UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Eugênio Simão

SERVIÇOS DIFERENCIADOS EM REDES IP:
diferenciação de serviços de rede utilizando um roteador CBQ.

Dissertação Submetida à Universidade Federal de Santa Catarina
como parte dos requisitos para obtenção do grau de
Mestre em Ciência da Computação

Prof. Roberto Willrich, Ph. D.

Florianópolis, março de 2002

SERVIÇOS DIFERENCIADOS EM REDES IP:
diferenciação de serviços de rede utilizando um roteador CBQ.

Eugênio Simão

Esta dissertação foi julgada adequada para obtenção do título de **Mestre em Ciência da Computação** na Área de Concentração de **Sistemas de Computação** e aprovada em sua forma final pelo Programa de Pós Graduação em Ciência da Computação.

Prof. Fernando A. O. Gaulthier, Ph. Dr.

Banca Examinadora:

Prof. Roberto Willrich, Dr.
Orientador

Prof. Vitorio B. Mazzola, Dr.

Prof. João Bosco Mangueira Sobral, Dr.

Relógio! deus sinistro, hediondo, indiferente,
Que nos aponta o dedo em riste e diz: "Recorda"
A dor vibrante que a alma em pânico te acorda
Como num alvo há de encravar-se brevemente.

(*As flores do mal*, Charles Baudelaire.)

Aos meus pais ...
Emílio e Clara Simão

Agradecimentos

Agradeço a Deus pela oportunidade e a seu filho pela possibilidade de ter como meus pais Emílio e Clara, por sua perseverança em me apontar o caminho dos estudos.
Agradeço a minha família, Luciane, Pedro e Rodrigo, por sua paciência e compreensão ao me ajudarem neste trabalho com seus sorrisos e brincadeiras.

Agradeço ao meu orientador Roberto Willrich, por sua obstinada compreensão e paciência quanto à realização deste trabalho.

Agradeço ao pessoal do provedor CNX por sua colaboração quanto à aplicação de métodos e de levantamento de dados estatísticos, em especial a Marcos Dias Assunção.

Resumo da Dissertação apresentada a Universidade Federal de Santa Catarina como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciência da Computação.

SERVIÇOS DIFERENCIADOS EM REDES IP: diferenciação de serviços de rede utilizando um roteador CBQ.

Eugênio Simão

Março de 2002

Orientador: Prof. Roberto Willrich, Ph. D.
Área de concentração: Sistemas de Computação.
Palavras-chave: **Qualidade de Serviço, Serviços Diferenciados, Internet.**
Número de páginas: 132.

A busca por modelos de diferenciação de serviços sobre o modelo Internet tem produzido diversas arquiteturas, todas as quais tem como objetivo em comum: tornar este modelo *gerenciável*, *atrativo* e *lucrativo*. Diferenciar significa, no entanto, correr o risco de produzir soluções que possam induzir ao modelo Internet, características contrárias às propriedades fundamentais que fizeram do modelo IP um modelo de sucesso. Este modelo tem como propriedade fundamental exibir um comportamento *imparcial, adaptativo* e *escalável*. Mesmo que soluções algorítmicas para a diferenciação de serviços acabem por induzir uma sobrecarga de processamento, estas se mostram flexíveis e capazes de atingir este objetivo, ao mesmo tempo em que preservam os requisitos de imparcialidade e adaptabilidade sobre a partilha de recursos de rede. Este trabalho busca a investigação destas soluções, mais precisamente aquelas que se referem ao compartilhamento hierárquico de recursos de rede. Através da utilização de um escalonador CQB, procura-se investigar o comportamento do modelo IP, quando a este for submetido uma estrutura hierárquica de controle de tráfego, a qual será responsável pela divisão do canal de comunicação em classes de serviço de redes.

Abstract of Dissertation presented to Universidade Federal de Santa Catarina
as a partial fulfillment of the requirements for the degree of
Master in Computer Science.

SERVIÇOS DIFERENCIADOS EM REDES IP:
diferenciação de serviços de rede utilizando um roteador CBQ.

Eugênio Simão

Março de 2002

Advisor: Prof. Roberto Willrich, Ph. D.
Concentration area: Computational Systems.
Keywords: **Quality of Services, Differentiated Services, Internet.**
Number of pages: 144.

A lot of efforts have been made by researchers and computers scientists to obtain a differentiated service model over the traditional Internet model. All of these efforts have a point in common: to make this model *manageable*, *attractive* and *profitable*. However, differentiate means get the risk of creating new models that don't take into account the fundamental proprieties that made the IP model a success: *fairness* and *adaptability*. While inducting some complexity into algorithmically solutions and producing some processing overhead, these seams to be flexible and capable in preserving the fundamentals proprieties at the same time that differentiation are achieved. In this work we investigate some of this algorithms, specifically those related to hierarchical link sharing, that forms the base to differentiated services at the level of mechanisms. By implementing a CBQ router we expect to observe the behavior of the IP model when a hierarchical set of rules are applied.

Sumário

Lista de figuras

Capítulo 1

Introdução

O desenvolvimento de modelos que permitam a construção de redes multi-serviços tem gerado algumas proposições e arquiteturas. O consenso, no entanto, é que em nenhum caso deva ser desconsiderado o modelo IP. O sucesso deste modelo deve-se ao seu comportamento escalável, imparcial e adaptativo. Sendo assim, qualquer modelo que tenha como objetivo promover diferenciações entre serviços de rede, fatalmente irá impor restrições a estes comportamentos fundamentais.

Soluções apresentadas para o problema de diversificação de serviços de rede, basicamente consistem em dois métodos: um que utiliza comutações de circuitos dedicados às exigências de cada serviço e outro que procura o mesmo resultado através da introdução de novos algoritmos de multiplexação estatística.

A simulação de canais dedicados através da utilização de algoritmos de multiplexação estatística apresenta-se como uma alternativa capaz de produzir a diferenciação de serviços enquanto preserva as propriedades fundamentais do modelo IP. Devido à flexibilidade inerente a soluções realizadas por software, pode-se adotar políticas de utilização de recursos de rede entre estes canais, que também demonstrem um caráter imparcial, adaptativo e escalável.

Portanto, este trabalho procura investigar o comportamento do modelo IP quando submetido a algoritmos de divisão do canal de comunicação físico em

sub-canais que demonstrem comportamentos específicos em termo de fatores de atraso, variação de atraso, perda de pacotes e prioridades relativas entre estes canais. Este objetivo constitui na verdade o objetivo do modelo de Serviços Diferenciados, onde o comportamento específico exibido pelo fluxo de pacotes que trafegam em cada um destes sub-canais, em cada nodo da rede, constitui na verdade uma classe de serviço de rede, que na terminologia deste modelo é denominado de PHB, ou *Per-Hop-Behavior*.

No momento de construção deste trabalho foram identificados os seguintes trabalhos relacionados:

"Qualidade de Serviços em Redes de Comutação de Pacotes" por Rui Pedro Magalhães Claro Prior, Faculdade de Engenharia da Universidade do Porto, março de 2001.

"Serviços Diferenciados em Redes IP: medições e testes para aplicações envolvendo mídias contínuas" por Renato Donizete Vilela de Oliveira, Universidade Federal de Santa Catarina, abril de 2001.

Sendo que o primeiro procura dar uma abordagem mais ampla aos serviços disponíveis no ambiente Linux para produzir qualidade de serviço em redes de computadores. O segundo, procura criar classes de serviço para atender a serviços multimídia, também utilizando o ambiente Linux como roteador com qualidade de serviço.

1.1 Objetivos do Trabalho

Se o objetivo da construção de uma rede multi-serviço é o de torná-la gerenciável, atrativa e lucrativa, então diferenciar serviços de rede implica também em diferenciar as expectativas dos prováveis clientes destes serviços. Desta forma, argumenta-se que devam existir três níveis de abstração distintos sobre serviços de rede, assim denominados: Classes de Serviços de Rede ao Cliente, Classes Abstratas de Serviços de Rede e Mecanismos de Rede.

- O isolamento do nível de **Classes de Serviços de Rede ao Cliente** quanto a sua percepção sobre o que para ele represente um serviço de rede, permite que ações de gerenciamento de expectativas sejam aplicadas de forma a capturar o comportamento exigido pelo cliente sobre um serviço de rede.

- As expectativas extraídas do nível de abstração do cliente poderão ser encapsuladas na forma de uma **Classe Abstrata de Serviços de Rede**, a qual deverá preservar o comportamento esperado pelo cliente

em uma classe de serviço, sendo ao mesmo tempo independente de sua implementação.

- A realização do comportamento capturado por uma classe abstrata de serviços de rede poderá ser efetuada através da utilização de algoritmos, ou **Mecanismos de Rede**, os quais serão responsáveis pela implementação do comportamento esperado para a classe de serviço.

Os objetivos deste trabalho estão constituídos pelos seguintes aspectos: o de investigação da existência destes três níveis de abstrações e o de experimentação de um mecanismo de diferenciação de serviços de rede para uma situação real.

O cenário desta situação real será constituído pelo ambiente de um provedor de acesso a Internet comercial, sobre o qual irá procurar-se explorar, em toda a sua extensão, os conceitos, fundamentos e aplicabilidade resultantes desta investigação. Sendo seus elementos chaves:

- **O modelo de Serviços Diferenciados**, o qual constitui a base e o alvo principal de todos os objetivos. Um estudo e uma reflexão sobre os fundamentos deste modelo deverão colaborar para a definição própria dos níveis de abstrações atribuídos aos serviços de rede.

- **As classes de serviços de rede**, as quais deverão evidenciar um isolamento próprio de cada nível de abstração, formas de relacionamentos entre estes níveis e realizações possíveis na forma de classes de serviços de rede efetivas.

- **O escalonador de compartilhamento** do canal de comunicação, o qual deverá permitir a construção de serviços distintos sobre este canal utilizando-se de multiplexação estatística. Serão utilizados os escalonadores CBQ, *Class-Based Queuing* e o HTB, *Hierarchical Token Bucket,* para este objetivo, devido à aplicabilidade que estes apresentam.

- O ambiente do sistema operacional Linux, o qual irá constituir **o roteador CBQ** em sua ultima instância. Devido ao conjunto de ferramentas que este sistema dispõe para o controle e condicionamento de tráfego, bem como, implementações dos escalonadores CBQ e HTB.

1.2 Organização do Texto

O segundo capítulo introduz o modelo de Serviços Diferenciados, seus fundamentos e terminologias, com o objetivo de identificar neste modelo base teórica suficiente para a definição dos níveis de abstrações de serviços de rede.

O terceiro capítulo descreve as inter-relações entre os níveis de abstrações de serviços de rede, bem como, a localização de classes de serviços sobre modelos de serviços, definidos em função da natureza dinâmica das relações contratuais entre cliente e provedor de serviço.

O quarto capítulo desenvolve o conceito do terceiro nível de abstração, ou seja, mecanismos de rede, sendo estes distinguidos em duas categorias: mecanismos de condicionamento de tráfego e mecanismos de diferenciação de serviços.

O quinto capítulo expõe os fundamentos de dois mecanismos de diferenciação de serviços de rede: o mecanismo CBQ (*Class-Based Queueing)* e HTB (*Hierarchical Token Bucket).*

O sexto capítulo expõe o conjunto de ferramentas de software e a arquitetura de controle de tráfego disponível no sistema operacional Linux, bem como, o ambiente de aplicação do algoritmo de diferenciação de serviços. Também será feita a análise dos resultados obtidos nos experimentos, comparando o comportamento do modelo IP, quando forem aplicados os algoritmos de diferenciação de serviços CBQ e HTB.

O sétimo capítulo refere-se diretamente a conclusão deste trabalho.

Os apêndices A e B apresentam arquivos de configuração das interfaces e descrevem os métodos de tomadas de dados dos experimentos.

Capítulo 2

Serviços Diferenciados

A Internet atualmente fornece serviços de rede, todos sujeitos a um único tipo de comportamento, denominado de melhor esforço: o tráfego é tratado tão rápido quanto possível, mas não há garantias temporais ou de limites de erro. O termo serviço é usado para descrever funções internas da rede como são percebidas pelos usuários finais, tais como aquelas que ocorrem nas comunicações fim-a-fim ou em aplicações cliente-servidor.

Com a rápida transformação da Internet em uma infra-estrutura comercial, o fornecimento de qualidade de serviço está sendo considerado cada vez mais um requisito essencial. O casamento dos termos, qualidade e serviço, pode ser visto como a capacidade de diferenciar tipos de tráfego, ou tipos de serviço, de forma que o sistema possa tratar uma ou mais classes de tráfego diferentemente de outras classes.

Neste sentido, seria interessante que um provedor de serviços Internet pudesse fornecer a seus clientes diversas classes de serviços, diferenciadas entre si por parâmetros de qualidade de serviço, geralmente dados em termos do atraso, variações no atraso, perda de pacotes e prioridades relativas entre classes. Sendo natural que a opção por uma classe de serviços que forneça garantias mais rígidas em termos destes parâmetros, tenham um custo maior

do que aquele pago por um tipo de serviço melhor esforço. Por exemplo, uma classe de serviço forneceria serviços Internet "previsíveis" para companhias que fazem negócios na Web. Tais companhias têm interesse em pagar um certo preço para tornar seus serviços confiáveis e fornecer a seus usuários um acesso rápido a seus sites Web. Estas classes poderiam ser constituídas por um único serviço ou por uma categoria de serviços classificados em termos da qualidade relativa entre eles, tal como sugere as classes *Ouro, Prata e Bronze*.

Outras classes de serviço poderiam ainda fornecer serviços que exibam atraso mínimo e baixa variação de atraso para aplicações tais como telefonia Internet e videoconferência. Companhias poderão optar em pagar um determinado preço para executar uma videoconferência de alta qualidade para reduzir custos e tempo de trabalho, através da utilização de ambientes colaborativos. Finalmente, o Serviço Melhor Esforço permanecerá para clientes que requerem apenas conectividade.

Este capítulo introduz os fundamentos relacionados ao modelo IETF[1] de Serviços Diferenciados, descrito basicamente pelos documentos da RFC 2474 e RFC 2475. No primeiro documento estão relacionados principalmente a definição do campo DS[2] de um pacote IP e sua codificação possível, definições de escopo das atribuições pertinentes ao grupo de trabalho, denominado de DiffServ, seus objetivos e definição da terminologia relacionada ao domínio do problema. No segundo documento estão definidos basicamente a arquitetura proposta para o modelo de Serviços Diferenciados, seus principais elementos e funcionalidades esperadas de cada elemento no sistema. Ainda, este capítulo, no contexto deste trabalho, tem como função principal evidenciar os elementos estruturais presentes em uma rede multi-serviço: Classes de Serviços ao Cliente, Classes Abstratas de Serviço de Rede e Classes de Serviços de Rede efetivas. Sendo esta última implementada na forma de Mecanismos de Rede. Desta forma, a leitura e interpretação feita dos documentos base gerados pelo grupo DiffServ está motivada segundo este propósito.

2.1 O Modelo Serviços Diferenciados

Diferenciar serviços de rede sobre o modelo IP, significa procurar por metodologias que permitam que a rede responda com um comportamento diferenciado em função de sinalizações distintas aplicadas sobre o conjunto de dados que trafegam sobre a rede. Basicamente existem duas alternativas:

---

[1] *Internet Engineering Task Force.*

[2] A abreviatura DS será utilizada neste trabalho para designar o modelo *Differentiated Services*, bem como para fazer referência ao campo de mesmo nome do pacote IP, definido por este modelo.

- A primeira não considera a existência de nenhuma sinalização interna aos pacotes de dados que trafegam sobre a rede e sim o canal de comunicação ao qual os pacotes são submetidos é que determinará o comportamento destes fluxos de pacotes.

- A segunda alternativa admite que a sinalização de diferenciação entre fluxos de pacotes esta contida no próprio pacote, permitindo que pacotes sejam encaminhados e constituam por si próprios canais de comunicação distintos em função da sinalização contida internamente ao pacote.

O modelo de Serviços Diferenciados adota esta segunda alternativa e considera que o meio de comunicação é determinado pela própria mensagem (o meio é a mensagem). Para tal foi definido um campo interno ao pacote IP denominado de campo DS, ou *Differentiated Service Field.* A codificação atribuída a este campo em princípio é livre e segue de certa forma a mesma filosofia dos endereços IP. Ou seja, um endereço IP é um endereço abstrato e pode ser mapeado para qualquer endereço físico, permitindo desta forma uma independência das tecnologias da camada física de rede. Da mesma forma, uma codificação contida no campo DS de um pacote é uma codificação abstrata do comportamento esperado por este pacote, permitindo uma independência do mecanismo interno de rede que irá implementar este comportamento.

Esta decisão simples permitiu que a complexidade inerente à manutenção de estado do valor semântico de uma marcação ao longo do percurso de uma conexão fosse "quebrada" em partes, pois à marcação somente será atribuído um valor semântico no escopo local de um domínio de rede e não ao longo de seu percurso. Ou seja, a flexibilidade introduzida por esta decisão permite que diversas interpretações sobre o que seja qualidade de serviço sejam tomadas no escopo local de um domínio de rede. Interpretações sobre o mesmo nível de qualidade de serviços entre domínios de rede deverão ser asseguradas por relações contratuais, desvinculando este tipo de informação do próprio sistema da rede e contribuindo para um decréscimo da complexidade inerente a manutenção destas informações em cada nodo da rede.

A fronteira que possa existir entre domínios de rede, na qual serão tomadas decisões de interpretação sobre qualidade de serviço, está claramente delineada no próprio abstract da RFC 2475 [Blake et al, 1998]: Existe uma distinção clara entre as funções que devam ser atribuídas aos nodos de borda e aos nodos interiores de uma rede. Aos nodos de borda é atribuída a responsabilidade de marcação (remarcação) dos bits no campo DS de um pacote, bem como, executar funções de condicionamento de tráfego. Aos nodos interiores cabe a função de encaminhamento de pacotes para seus diversos destinos em função do valor semântico atribuído a marca.

Provedores de serviços podem então construir diferentes serviços de rede a partir dos seguintes blocos fundamentais: *marcação* do campo DS, *condicionamento* do tráfego e *encaminhamento* de pacotes. Ainda, para que estes serviços possam demonstrar um comportamento consistente faz se necessário à adoção de regras nas fronteiras de uma rede. Estas regras, dizem respeito a como os bits do campo DS devam ser marcados, como os pacotes devam ser condicionados na fronteira da rede e como os pacotes devam ser encaminhados para o interior da rede, como ilustra a Figura 2.1.

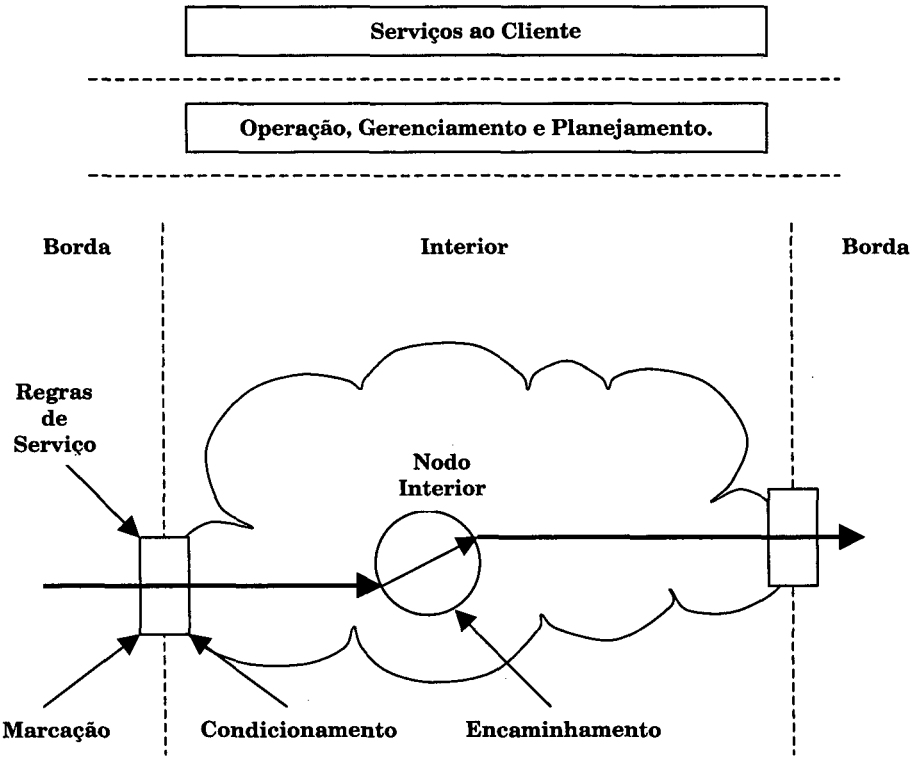

Figura 2.1 Blocos fundamentais que definem um domínio de rede segundo o modelo de Serviços Diferenciados.

2.1.1 Arquitetura dos Serviços Diferenciados

A RFC 2475, "An Architecture for Differentiated Services", define a arquitetura para diferenciação de serviços, escaláveis, com base na especificação do campo DS [Black et al. 1998]. Devido ao fato de que a arquitetura não é ela mesma uma questão a ser padronizada, esta possui o status de documento informacional junto a IETF.

Os elementos básicos da arquitetura do modelo de Serviços Diferenciados são explicados no segundo capítulo da RFC 2475, "An Architecture for Differentiated Service". Os elementos chaves desta arquitetura recaem sobre

as seguintes definições: *nodos de borda*, *nodos interiores*, *nodos de ingresso* e *nodos de egresso*, como ilustrados pela Figura 2.2. Estes elementos, como entidades próprias, são de certa forma virtuais, pois todas estas características podem ser atribuídas a um único nodo da rede.

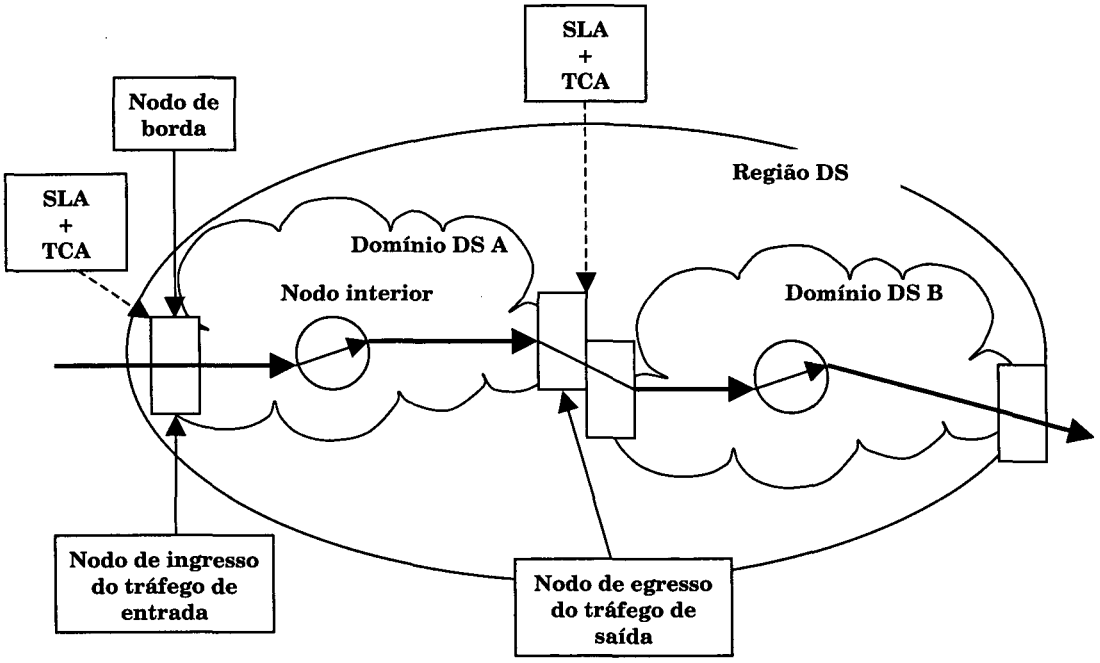

Figura 2.2 Elementos básicos da arquitetura do modelo de Serviços Diferenciados.

Desta forma, pode-se dizer que um nodo da rede pode ser descrito por um conjunto de funções características:

Nodo de borda: Deve apresentar um conjunto de funções necessárias a interconectividade entre domínios DS, bem como a domínios que não sejam compatíveis ao modelo DS.

Nodo interior: Deve apresentar um conjunto de funções necessárias a conectividade com demais nodos que sejam compatíveis ao modelo DS.

Nodo de ingresso: Deve apresentar um conjunto de funções necessárias ao controle de tráfego geralmente empregado aos fluxos que dão entrada a um domínio DS.

Nodo de egresso: Deve apresentar um conjunto de funções necessárias ao controle de tráfego geralmente aplicado aos fluxos que deixam um domínio DS.

Entre domínios de rede que sejam compatíveis ao modelo DS, os requisitos dos modelos de serviços aos clientes e o respectivo mapeamento destes requisitos para os serviços internos de rede, são feitos em duas formas contratuais:

Service-Level Agreement (SLA): Define um contrato entre um cliente e um provedor de serviços, o qual especifica o serviço de encaminhamento que um cliente deva receber. Onde um cliente poderá ser uma organização (fonte do tráfego) ou um outro domínio compatível ao modelo DS (destino do tráfego). Um SLA poderá incluir regras de condicionamento de tráfego, as quais poderão constituir um TCA no todo ou em parte.

Traffic-Conditioning Agreement (TCA): Define um contrato que especifica as regras de classificação e os perfis de tráfegos correspondentes, bem como, as regras de medição, marcação, descarte e conformação, as quais deverão ser aplicadas aos fluxos selecionados pelo classificador. Um TCA engloba todas as regras de condicionamento de tráfego explicitamente especificadas em um SLA conjuntamente as regras implícitas aos requisitos relevantes de um serviço e/ou a partir das políticas de provisão de serviços de um domínio DS.

Um domínio DS é uma parte de uma rede na qual os nodos desta rede são compatíveis ao modelo DS, sendo que sobre esta rede são aplicados grupos de PHBs baseados pelo mesmo mecanismo e políticas de provisão de serviços. Na interface entre domínios, onde provavelmente haverá mudanças na estrutura de PHBs aplicada, bem como, nas políticas de provisão de serviços é necessário que exista um nodo de borda, o qual fará o mapeamento entre PHBs de cada domínio. Um conjunto contíguo de domínios DS forma uma região DS. As diretrizes previstas no modelo de Serviços Diferenciados podem ser aplicadas sobre uma região DS, mas mudanças significativas que possam ocorrer entre os PHBs de cada domínio pode tornar difícil projetar e implementar serviços fim-a-fim. Soluções para este tipo de problema foram introduzidas recentemente pela RFC 3086, a qual procura determinar o comportamento esperado de um fluxo de pacotes entre domínios de rede, *Per-Domain Behavior (PDB)* [Nichols e Carpenter, 2001].

2.1.2 Classificação e Condicionamento de Tráfego

A Figura 2.3 apresenta a estrutura lógica de classificação de tráfego, bem como funções de condicionamento de tráfego. Esta estrutura baseia-se na premissa de que a classificação é feita com base em informações contidas no cabeçalho dos pacotes que chegam pela interface de entrada, como descrito na RFC 2474 [Nichols et al. 1998]. Este modelo sugere que o mecanismo de medição não tenha nenhum efeito sobre a classificação. Este é um modelo

viável supondo-se que classificar signifique selecionar uma classe PHB ao invés de um PHB individual. Neste caso, marcação é uma ação tomada dentro de uma classe e não ultrapassa a fronteira entre classes de PHBs.

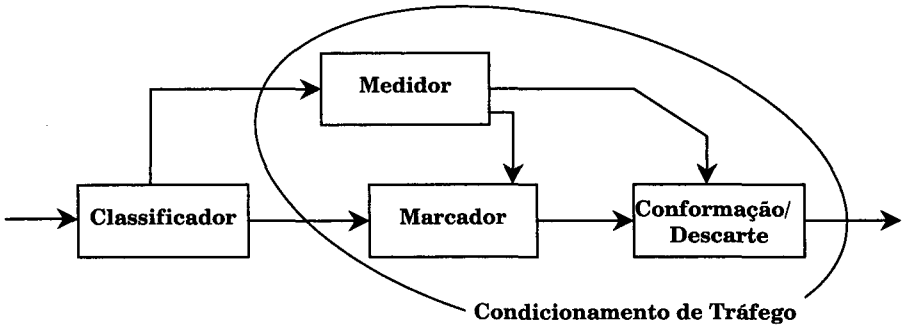

Figura 2.3 Classificação e condicionamento de tráfego de acordo com a RFC 2475.

As regras de condicionamento de tráfego podem ser impostas baseando se no *perfil* de cada tráfego. Em um modelo simples cada pacote poderá estar em *conformidade ao perfil* ou *fora do perfil*. Pacotes que estejam conformes ao perfil obtém melhores tratamentos de condicionamento e de encaminhamento do que pacotes que estejam fora do perfil.

2.1.2.1 Medição

De acordo com a RFC 2475, o *medidor* é responsável por efetuar uma verificação do aporte de fluxo de pacotes em cada classe. O medidor então notifica aos mecanismos de marcação, conformação e descarte sobre o estado deste fluxo.

2.1.2.2 Marcação

Este mecanismo tem por função marcar o codepoint apropriado ao campo DS de um pacote. Isto significa que o marcador também poderá modificar o valor original contido neste campo. De maneira a evitar a reordenação internamente a rede, o marcador deverá obedecer determinadas regras quando for modificar uma marcação de um pacote.

2.1.2.3 Conformação

Este mecanismo é utilizado para adequar o comportamento bruto demonstrado pelo aporte de um fluxo de pacotes ao comportamento previsto para o perfil da classe, efetuando desta forma, uma conformação do aporte de tráfego.

2.1.2.4 Descarte

Este mecanismo verifica a condição de um pacote tomando por base políticas de descarte descritas nos contratos SLA e TCA, sendo que o mecanismo de condicionamento de tráfego poderá então decidir sobre os níveis de prioridade de descarte deste pacote.

2.1.2.5 Condicionamento

Como ilustrado pela Figura 2.3, o condicionamento de tráfego é executado com base nas informações compostas pelos mecanismos de medição, marcação, conformação e descarte. O condicionamento de tráfego está geralmente localizado em nodos de borda compatíveis ao modelo DS. A regra básica é que ações de classificação e de condicionamento sejam efetuadas próximas a fonte do tráfego.

2.2 Terminologias

Dentre as terminologias colocadas nas RFC destacam-se aquelas que introduzem os conceitos de: *Per-Hop Behavior*, *Serviços ao Cliente*, *Serviços de Rede*, *Classe PHB*, *Codepoint* e *Mecanismo*. Também devido a referência constante aos termos *fluxo* e *agregado*, os quais possuem significados similares, será introduzido as definições da RFC 2474 para *agregado* e *microfluxo*. Em outros casos o termo fluxo, quando utilizado, possui a conotação dada ao termo microfluxo.

2.2.1 Serviços

A RFC 2474 ao introduzir o conceito de serviço não faz distinção direta entre serviços ao cliente e serviços de rede. Serviços ao cliente são uma descrição do comportamento global esperado, pelo cliente, sobre o tratamento dado ao tráfego originado por ele (por uma aplicação que utilize). Enquanto que serviço de rede refere-se a sub-conjuntos de tratamentos possíveis, dados ao tráfego do cliente, os quais devem permitir a implementação do comportamento esperado pelo cliente.

2.2.2 Comportamento agregado

O termo comportamento agregado refere-se a um conjunto de pacotes que possuem a mesma marcação, ou codepoint, quando estes cruzam o canal de

comunicação em uma determinada direção. Sendo que a utilização do termo nas formas "agregado" e "comportamento agregado" são intercambiáveis.

2.2.3 Microfluxo

Um microfluxo compreende uma única instância do fluxo de pacotes originado entre aplicações, o qual poderá ser identificado pelo endereço de origem, endereço de destino, identificação do protocolo, porta de origem e porta de destino, quando aplicável.

2.2.4 Per-Hop Behavior

O termo *Per-Hop Behavior*, ou PHB, é o mais difícil de ser compreendido, mas também constitui o elemento fundamental para a idéia geral contida no modelo de Serviços Diferenciados. Do ponto de vista técnico o termo PHB possui a conotação combinada de diversas ações que devam ser tomadas em cada *hop*[3]: *encaminhamento*, *classificação*, *escalonamento* e comportamento de *descarte* de pacotes. Mas, este termo não agrega somente um significado técnico e sim comporta o significado de ser uma camada de abstração entre os modelos de serviços possíveis aos clientes e as implementações possíveis destes modelos na forma de serviços de rede. Baseando-se nas definições da RFC 2474, pode-se derivar as seguintes interpretações sobre um PHB de maneira a se obter linhas gerais para construção de novos PHBs:

- Um PHB é primeiramente uma descrição do comportamento esperado, para um serviço de rede, em um nível de abstração elevado.
- Um PHB deverá permitir a construção de serviços de comportamento previsíveis.
- O comportamento esperado deverá ser externamente observável e a descrição do comportamento não deverá fazer referências de seus elementos internos, tal como fila.
- O comportamento esperado deverá ter significado local, ou seja, o significado deve estar associado ao nodo local ao invés de toda a rede.
- A descrição do comportamento está relacionada a características exigidas por um agregado, o qual consiste em todos os pacotes que pertençam ao um mesmo PHB, em um certo ponto da rede.
- Os pacotes que pertençam a um mesmo PHB deverão experimentar o mesmo tipo de tratamento independente de outras informações contidas

[3] Um *hop* é um *lance* entre nodos de uma rede nos quais ocorrem decisões de roteamento e encaminhamento de pacotes.

no pacote e independente dos fluxos individuais que possam existir junto ao agregado.

- A descrição de um PHB não deverá pressupor nenhuma forma de condicionamento de tráfego na fronteira da rede.

Destas observações, as duas primeiras constituem o objetivo geral dado a um PHB, ou seja, este deve proporcionar uma base sólida para o entendimento do comportamento de um sistema que adote as diretrizes do modelo de Serviços Diferenciados. As demais interpretações limitam o termo PHB de maneira que possa ser restringido ao comportamento esperado. A Figura 2.4 ilustra um modelo simplificado de especificação de um PHB, na forma de uma caixa preta, para um nodo interior da rede, o qual está relacionado ao tratamento de agregados.

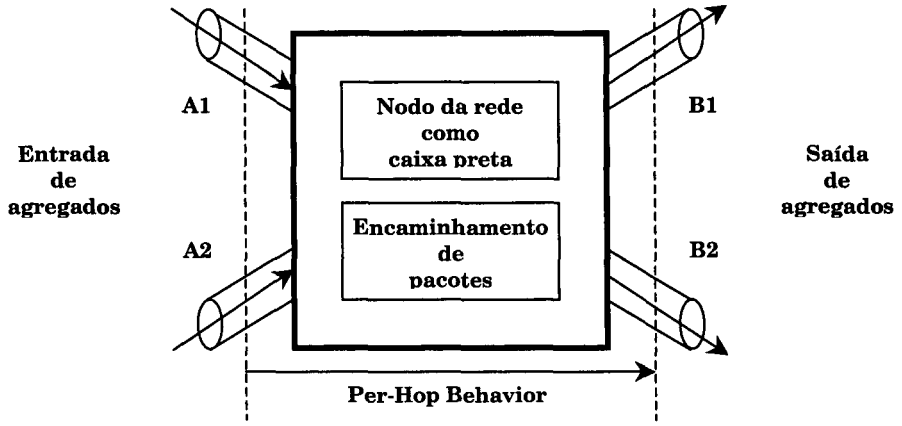

Figura 2.4 Modelo simplificado de um PHB para um nodo interior da rede.

A última interpretação observada está na verdade descrita na RFC 2475, a qual explicitamente declara que a arquitetura de Serviços Diferenciados deverá desacoplar condicionamento de tráfego e funções de provisão de recursos dos comportamentos de encaminhamento. A principal razão para o desacoplamento entre condicionamento de tráfego e comportamentos de encaminhamento é a flexibilidade. Pois, funções de condicionamento de tráfego evoluem independente da arquitetura do modelo de Serviços Diferenciados.

2.2.5 Classe PHB

Ainda que não esteja explicitamente declarado na RFC 2474, o termo *classe* PHB tem sido utilizado em especificações de PHBs, especificamente em [Baker et al. 1998]. Uma *classe* PHB é uma coleção de PHBs criados com o objetivo de transmitir pacotes de uma aplicação. Isto significa que o provedor

de serviços poderá remarcar pacotes dentro de uma mesma classe, mas não poderá remarcar de uma classe para outra. O requisito principal de uma classe PHB é que pacotes não devam ser reordenados internamente a rede. Uma classe PHB, dado que exista uma função apropriada de condicionamento de tráfego aplicada na fronteira da rede, constitui o equivalente aos serviços orientados a conexão como ocorre para redes ATM[4] [Kilkki, 1999].

2.2.6 Mecanismo

De acordo com a RFC 2474, um *mecanismo* é a implementação de um PHB de acordo com um algoritmo em particular. Um mesmo algoritmo pode ser utilizado para implementar diversos PHBs e, de outra forma, muitos mecanismos podem implementar um único PHB.

2.2.7 Codepoints

Codepoints são indicadores utilizados internamente a rede para informar aos nodos da rede o PHB apropriado de um pacote. O requisito fundamental de um codepoint é que este deverá identificar de forma não-ambígua o PHB a qual pertence. Também, diversos codepoints poderão ser mapeados para um mesmo PHB, significando que um agregado pode ser composto por fluxos que possuem codepoints distintos. Neste caso, o tratamento dado ao agregado deverá ser o mesmo previsto para o PHB, para o qual os pacotes foram mapeados e independente da marcação contida no pacote.

2.3 Definição do Campo DS

A informação sobre um determinado PHB é transmitida no campo denominado de *Differentiated Services Field*, ou campo DS, de um pacote IP. A seção 3 da RFC 2474 define a estrutura deste campo, ilustrada pela Figura 2.5. Esta nova definição substitui definições anteriores dadas a este campo, sendo que para o protocolo IPv4 substitui a definição *Type of Service*, ou TOS. Enquanto que para o protocolo IPv6 substitui a definição *Traffic Class*. O octeto que compõe o campo DS é dividido em duas partes: os primeiros seis bits compõe o campo DSCP, ou *Differentiated Services Codepoint*, o qual é utilizado para selecionar o PHB ao qual o pacote pertença e os últimos dois bits são utilizados como reserva futura.

[4] Asynchronous Transfer Mode, ou redes ATM, onde os serviços de rede estão baseados ao equivalente a comutação de circuitos, donde a denominação de serviços orientados a conexão.

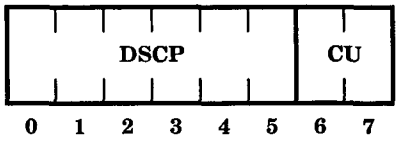

Figura 2.5 Estrutura do campo DS contido no cabeçalho de um pacote IP.

Um nodo da rede que seja compatível ao modelo de Serviços Diferenciados deverá utilizar todos os seis bits do campo DSCP e nenhum outro bit para selecionar um PHB. Ainda, a RFC 2474 requer que a implementação do mapeamento de codepoints seja realmente flexível: este campo deverá ser tratado como um índice, sem nenhuma estrutura interna predefinida, e o operador deverá ser capaz de mapear qualquer codepoint para qualquer PHB.

2.3.1 Definições históricas atribuídas ao Codepoint

Se um pacote contendo um codepoint irreconhecível for recebido, este deverá ser encaminhado de acordo com um comportamento default. Desta forma um PBH default deverá estar disponível em cada nodo compatível ao modelo DS. Este comportamento default corresponde ao serviço melhor esforço, sendo que este último diz respeito ao método original de encaminhamento de pacotes utilizado pelo modelo Internet. Ou seja, a rede tentará despachar tantos pacotes quanto forem possíveis tão breve quanto for possível.

A questão mais importante relacionada ao PHB default é a sua relação com outros PHBs utilizados na rede. De maneira geral é recomendado que o PHB default sempre obtenha recursos de rede em capacidades razoáveis, permitindo desta forma uma boa coexistência entre o modelo Internet original com o modelo de Serviços Diferenciados. Devido a esta reserva de compatibilidade o codepoint 000000 deverá mapear pacotes para o PHB default (ou para um outro PHB de mesma característica).

2.3.2 O PHB Class Selector

Além de definir um PHB default, a RFC 2474 define um grupo de PHBs denominado de *class selector* PHB, com codepoints padronizados. A razão para esta padronização é que o campo de *precedência* do protocolo IP definido na RFC 791 é realmente empregado em algumas redes, orientando desta forma uma compatibilidade de marcação entre estes modelos.

2.3.3 Diretrizes para padronização de PHBs

A RFC 2474 proporciona algumas diretrizes para aqueles que pretendem projetar especificações de PHBs. Primeiro, as regras da própria IETF deverão ser seguidas através das ações de especificação, implementação, aplicação e provas de utilidade, as quais compõem os pré-requisitos para a construção de qualquer PHB a ser padronizado.

Devido ao fato de que mecanismos não são padronizáveis, fornecedores poderão utilizar conjuntamente qualquer mecanismo que satisfaça a definição de um PHB ou classe de PHBs. Presume-se que alguns PHBs irão evoluir de tal maneira que novos conjuntos de serviços irão emergir a partir deles. Ainda, é prematuro afirmar qual especificação de PHB terá possibilidades de evoluir, ou qual proposta se tornará comum, e quais dentre elas irão desaparecer.

Ainda que por definição o campo DSCP não deveria possuir nenhuma atribuição pré-definida a RFC 2474, no capítulo 6, sugere três diferentes conjuntos, principalmente com o propósito de permitir o gerenciamento de codepoints. O primeiro conjunto, descrito pela combinação 'xxxxx0' será utilizado com o propósito de padronização. O segundo conjunto, descrito pela combinação 'xxxx11' será utilizado com propósitos experimentais e de uso local. Enquanto que o terceiro conjunto, descrito por 'xxxx10' será inicialmente utilizado para uso experimental e a qualquer momento poderá ser utilizado com propósito de padronizações.

2.4 O PHB como elemento para alocação de recursos

De acordo com a RFC 2475 na seção 2.4, um nodo da rede que seja compatível ao modelo DS, baseia-se na descrição do PHB para determinar a largura de banda que o agregado deste PHB deva utilizar. Por exemplo, um determinado PHB poderá receber uma cota mínima de $x\%$ da capacidade total do canal, por um intervalo de tempo razoável. Ainda, sugere que a necessidade de largura de banda expressa por um PHB poderá ser medida, de forma simples e imparcial, para uma variedade de condições de competição entre tráfegos (outros PHBs).

Em uma configuração mais complexa poderia ser garantido a um PHB uma cota de x% da capacidade do canal e que os recursos de banda excedente sejam compartilhados proporcionalmente, de maneira imparcial, entre outros PHBs.

Nas observações, da seção 2.4, existe uma referência circular a definição dos termos PHB e agregado. Sugerindo, desta forma, que um pode ser expresso

em função do outro. Como um agregado pode ser expresso em função de largura de banda, então também um PHB o será. Portanto, em um ponto da rede, haverá um conjunto de PHBs que estarão concorrendo pela capacidade do canal, cada qual impondo as exigências de tratamento descritas pelas características intrínsecas de seu comportamento, em termo de vazão, atraso, variação de atraso, perda e prioridades relativas.

2.5 Diretrizes para o projeto de PHBs

A RFC 2475 estabelece quinze instruções adicionais para o projeto de PHBs. De acordo com o primeiro item, um PHB deverá incluir recomendações sobre a codificação adotada para o codepoint. Ainda, a especificação de um PHB deverá incluir o seguinte:

- Uma apresentação do propósito geral do grupo de PHB
- Especificação das interações individuais de um determinado PHB com o seu grupo
- Restrições relativas à provisão de recursos se necessário. Por exemplo, quando a própria funcionalidade do PHB dependa de ações de condicionamento
- Uma declaração sobre se o grupo de PHB for considerado para uso geral ou local
- Uma declaração sobre quais circunstâncias um pacote deva ser remarcado internamente ao grupo de um PHB para outro PHB.

Outras recomendações para a especificação de um PHB dizem respeito a interação com PHBs existentes, tunelamento, requisitos de conformidade, segurança, impacto sobre as camadas superiores da pilha de protocolos e com a camada de enlace.

2.6 Questões de Interoperabilidade

Um nodo da rede que não seja compatível ao modelo DS, não irá interpretar corretamente o PHB utilizado em um domínio de rede compatível ao modelo DS. Um caso específico de um nodo não compatível ao modelo DS, e que de certa forma mantém um certo nível de interoperabilidade, refere-se ao nodo que adote o modelo da RFC 791, o qual baseia em níveis de precedência dados aos pacotes IP e é amparado pelo PHB class selector. Um nodo que seja totalmente não compatível ao modelo DS irá impor a sua capacidade de

processamento sobre a carga total de todos os PHBs, atuando diretamente sobre as características de serviço de cada classe. Uma solução possível seria a de adotar neste nodo no mínimo o PHB class selector.

Um cenário mais complexo refere-se a transmissão de tráfego compatíveis ao modelo DS a um domínio de rede incompatível a este modelo. A alternativa que parece mais apropriada é a de remarcar os pacotes no ponto de egresso para o PHB default, ou para o PHB class selector.

2.6.1.1 Fluxos Multicast

A RFC 2475 discute brevemente duas questões relacionadas a fluxos multicast. Primeiro, relativo ao consumo de recursos de rede entre pacotes multicast e unicast que possam apresentar-se à rede. Pois, pacotes multicast exigem a existência de recipientes no nodo de destino, suficientes para que não ocorra uma monopolização de recursos. Uma solução apontada seria a de que pacotes multicast utilizem marcações diferentes e PHBs diferentes aos dos pacotes unicast.

A segunda questão está relacionada com a situação para a qual um pacote multicast que chega para um nodo no seu ponto de ingresso poderá ser transmitido para diversos destinos, ou domínios de rede diferentes. Sendo que este caso impõe uma questão de difícil solução sobre a marcação a ser feita de maneira que este esteja em conformidade a todos os domínios de destino.

2.6.1.2 Segurança e Tunelamento

A questão de segurança mais relevante para ambos os documentos RFC 2474 e RFC 2475 refere-se ao método de ataque por negação de serviços, ou *denial-of-service*, bem como interações possíveis entre protocolos de segurança. Ataques podem ocorrer devido a que alguns usuários irão tentar modificar os valores atribuídos aos codepoints de seus pacotes, de maneira a se valerem do melhor tratamento dado a determinados PHBs, o que ao final irá constituir um ataque por negação de serviço, pois no pior caso irá exaurir os recursos para estes PHBs.

O fato de que nodos interiores baseiam-se diretamente nas marcações geradas por nodos de borda permite que internamente a um domínio de rede possa ser feita uma marcação que possua significado local ao domínio, o que poderia contribuir para uma solução a tentativas de ataque deste tipo.

A seção 6.2 da RFC 2475, declara considerações a serem feitas quanto a utilização do modelo DS conjuntamente ao modelo IPSec, o qual é utilizado para produzir canais virtuais, ou túneis sobre a Internet, e a interação entre estes modelos.

2.7 Elementos Estruturais

Da leitura feita sobre os documentos bases do modelo de Serviços Diferenciados, e mais especificamente pelo cenário descrito pela Figura 2.6, chegou-se as seguintes constatações:

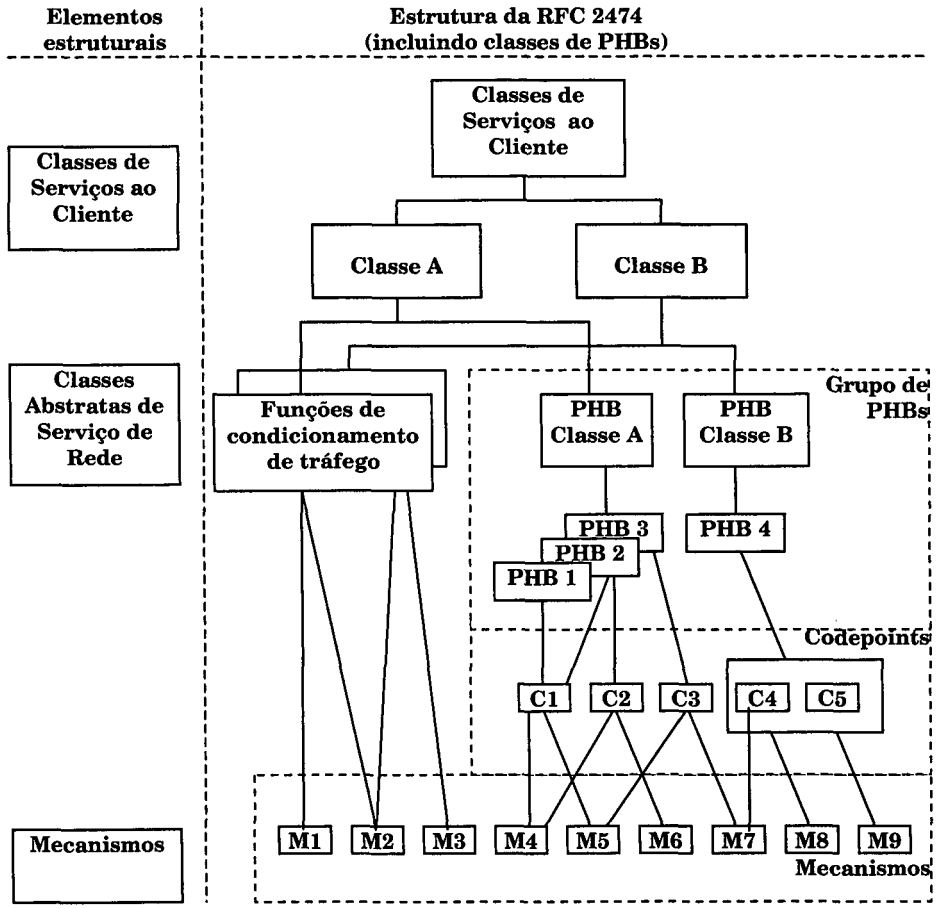

Figura 2.6 Relacionamento entre os elementos estruturais contidos nos documentos base do modelo de Serviços Diferenciados[5]

Uma referência intrínseca sobre três níveis de abstração de serviços de rede, os quais quando isolados produzem as seguintes classes de serviço: Classes de Serviços ao cliente, Classes Abstratas de Serviços de Rede e Mecanismos de Rede.

- O nível de abstração, **Classes de Serviços ao Cliente**, permite capturar a visão do cliente sobre serviços de rede e a conseqüente determinação de seus requisitos e características inerentes. Também, o seu isolamento, permite que campanhas de marketing façam uso deste nível de abstração

[5] Esta ilustração foi primeiramente apresentada por [Kilkki, 1999].

durante a divulgação do serviço referindo-se a um produto com características bem definidas e atrativas.

- O nível de abstração, **Classes Abstratas de Serviço de Rede**, é na verdade um encapsulamento da semântica definida para o termo PHB. O seu isolamento permite capturar os requisitos e características intrínsecas da classe de serviços ao cliente de maneira a determinar o comportamento previsto para esta classe. Ainda, pelo fato de ser abstrata é independente do mecanismo que deverá implementar este comportamento.

- O nível de abstração, **Mecanismo de Rede**, é na verdade a implementação do comportamento esperado para uma classe abstrata de serviços de rede. Sendo que esta ultima parametriza, através dos requisitos e características intrínsecas capturadas da classe de serviço ao cliente, a evolução ou o comportamento destes mecanismos.

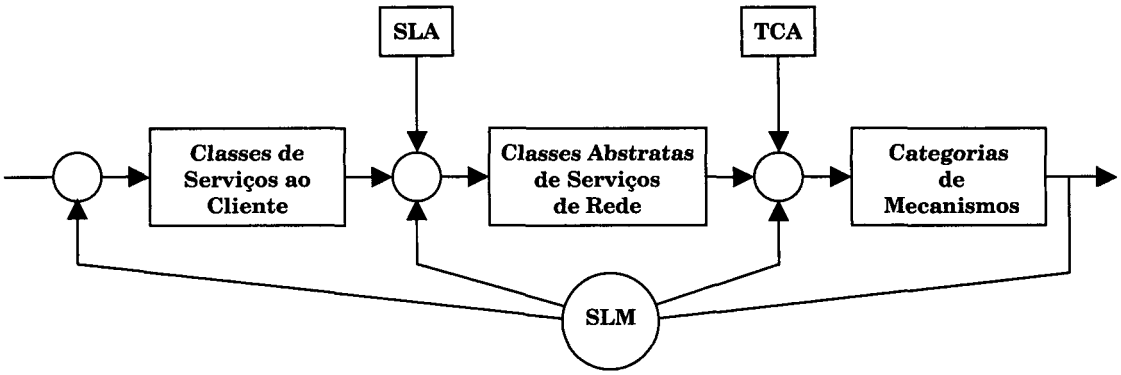

Figura 2.7 Mapeamento entre os diversos níveis de abstração de serviços de rede e os diversos níveis de gerenciamento de serviços de cada nível de abstração.

O mapeamento das informações entre estes três níveis de abstração se dá através de contratos de níveis de serviço. Sendo que o mapeamento entre a classe de serviços ao cliente e a classe abstrata de serviço de rede é feita através da adoção de um SLA. Enquanto que o mapeamento entre a classe abstrata de serviço de rede e um mecanismo ocorre através da descrição de um TCA. Permitindo que níveis de gerenciamento de serviços de rede, *Service Level Management (SLM)*, também possam ocorrer em suas diversas instâncias entre estes níveis de abstração de classes de serviço, como ilustra a Figura 2.7.

Portanto, o termo diferenciar, agora pode ser aplicado a estes níveis de abstração com objetivos distintos. Para obter-se uma diferenciação de serviços de rede também faz-se necessário a diferenciação de clientes e conseqüentemente de seus serviços. As interações entre estes níveis de abstração serão melhores desenvolvidas no capítulo 3.

2.8 Resumo

Este capítulo procurou evidenciar através da leitura dos documentos base do modelo de Serviços Diferenciados elementos fundamentais que contribuíssem para um melhor entendimento sobre a interpretação geralmente atribuída a diferenciação de serviços e as diversas conotações que suas instâncias acarretam.

Ficou constatado que a maior contribuição deste modelo foi a de constituir uma camada de abstração entre o cliente e a rede através da introdução do campo DS. Antes de ser um conjunto de normativas, este modelo é na verdade um modelo de referência para construção de classes abstratas de serviços de rede, PHBs. Também este modelo procura definir um domínio de rede com fronteiras bem definidas de forma a isolar o comportamento interno de um domínio de rede de outro. Inter-relações entre domínios de rede são dadas na forma de relações contratuais de maneira a permitir que a percepção de qualidade descrita em um domínio de rede possa encontrar o seu par correlato em outros domínios de rede. Contribuindo desta forma para o decréscimo da complexidade inerente a manutenção da mesma percepção de qualidade entre domínios de rede distintos, ou ao longo de um caminho fim-a-fim.

Esta flexibilidade contribui para que provedores de serviços criem os seus próprios "produtos de rede" em função das características intrínsecas dos serviços solicitados por seus clientes. Naturalmente haverá uma gama destes produtos que serão comuns a muitos domínios de rede, sendo que estes então deverão ser padronizados, na forma de PHBs, para aplicações interdomínios.

Capítulo 3

Classes de Serviços de Rede

Diferenciar serviços aos clientes implica em determinar um fator comum de discriminação de serviços. Este fator deverá preservar entre modelos distintos, características comuns, tais como: a *imparcialidade* quanto ao tratamento dado entre os diferentes serviços possíveis e a *consistência* de aplicação das regras de distinção. Estas características formam a base para políticas de formação de preços em função da qualidade do serviço oferecido, passíveis de serem compreendidas pelo cliente e conseqüentemente sugerem ao mesmo uma gama de escolhas possíveis quanto à classe de serviço que deseja adquirir.

Este capítulo introduz o aspecto dinâmico dos requisitos de qualidade, quando estes se apresentam a rede, como um fator comum de discriminação de serviços aos clientes, originando desta forma quatro modelos possíveis: *conexões dedicadas instantâneas*, *conexões dedicadas permanentes*, *conexões compartilhadas permanentes* e *conexões compartilhadas instantâneas* com precedência de qualidade.

3.1 Classes de Serviços ao Cliente

Classes de serviços ao cliente podem ser construídas tomando-se por base que o cliente pode situar-se em uma das seguintes categorias:

Usuário: Para a categoria usuário, assume-se que este determina as características exigidas para a classe de serviço, contratando uma capacidade em termos de largura de banda fixa junto ao provedor de serviços, a qual acredita ser suficiente para a execução de seus propósitos.

Organização: Na categoria organização, assume-se que uma organização adquire uma capacidade fixa, em termos de largura de banda e disponibiliza esta capacidade entre suas divisões. Neste caso, internamente a organização, os recursos de rede disponíveis são explorados por usuários e aplicações da melhor maneira possível.

Aplicação: O cliente, sob a perspectiva de uma aplicação, sugere uma relação dinâmica entre cliente e provedor de serviços. Onde, características distintas de cada aplicação são impostas sobre a rede de forma que esta deverá estar capacitada a atender requisitos distintos de serviço os quais apresentam uma variação dinâmica no tempo.

Entre estas categorias, a do usuário e a da organização fazem referência direta a capacidades fixas, ou dedicadas, em termos de largura de banda junto ao provedor de serviços. Sendo que solicitações feitas por uma organização apresentam um caráter mais permanente. Enquanto que, solicitações produzidas por um usuário apresentam um caráter mais dinâmico, quando comparada a uma organização e pode, menos freqüentemente, apresentar um comportamento dinâmico igual aquele exibido pela categoria das aplicações. Finalmente, solicitações feitas por aplicações, e seus requisitos, podem variar abruptamente no tempo. Desta forma, as relações contratuais entre estas entidades e o provedor de serviços, as quais têm como objetivo mapear os requisitos dos clientes para serviços de rede, também estarão submetidas à natureza dinâmica destas relações.

Serviços de rede quando solicitados em termos de largura de banda, como acontece para as categorias usuário e organização, possuem uma conotação quantitativa de qualidade. Pois estes adquirem uma quantidade de recursos, os quais devem manter esta característica de qualidade no tempo, independente das condições de outros clientes, constituindo desta forma serviços dedicados ou garantidos. Por outro lado, quando a decisão sobre a quantidade de recursos necessária para a realização de um determinado

serviço for atribuída ao sistema[6], este em função da sinalização de requisitos emitida, geralmente pela aplicação, pode decidir em oferecer recursos de rede compartilhados entre outras classes de serviço. Neste cenário, a qualidade de serviço assume uma conotação qualitativa ou relativa a outros serviços que estejam sendo compartilhados sobre um mesmo recurso físico, no tempo.

Desta forma, o caráter de qualidade associado a estas formas de solicitações pode determinar as seguintes categorias de solicitações de serviços:

Quantitativo: O caráter quantitativo está relacionado a solicitações de serviços que exigem uma quantidade dedicada (reservada) de recursos exigidos pelo cliente, independente da condição de outros clientes ou classes de serviços. Usuários e aplicações exploram esta quantidade da melhor maneira possível.

Qualitativo: Serviços baseados pelo caráter qualitativo são expressos por uma forma descritiva de seus requisitos intrínsecos, sendo que os recursos alocados são dados em função de parâmetros necessários a realização do serviço, o qual pode assumir aspectos de serviços dedicados (garantidos) ou de compartilhamento.

Relativo: O caráter relativo está associado a solicitações de serviços expressas em termos relativos a outras classes de serviço. Serviços são oferecidos da melhor maneira possível, quando comparáveis uns aos outros, em função da capacidade disponível de recursos.

Portanto, entre estas duas dimensões: dinâmica das relações contratuais e categorias de solicitações de serviços, descritas por seus caracteres quantitativo, qualitativo e relativo é possível de serem localizados modelos de serviços de rede. Como ilustra a Figura 3.1.

3.1.1 Modelos de Serviços

Modelos de serviços podem então ser localizados em função da natureza dinâmica das relações contratuais dispostas em relação a categorias de solicitações de serviços. Solicitações de serviços de caráter mais permanente, geralmente são executadas por agentes humanos. Enquanto que, solicitações dinâmicas por serviços podem ser efetuadas por agentes autônomos.

[6] Neste caso, sistema, refere-se a mecanismos internos a própria rede como também políticas internas de gerenciamento adotadas pelo provedor de serviços.

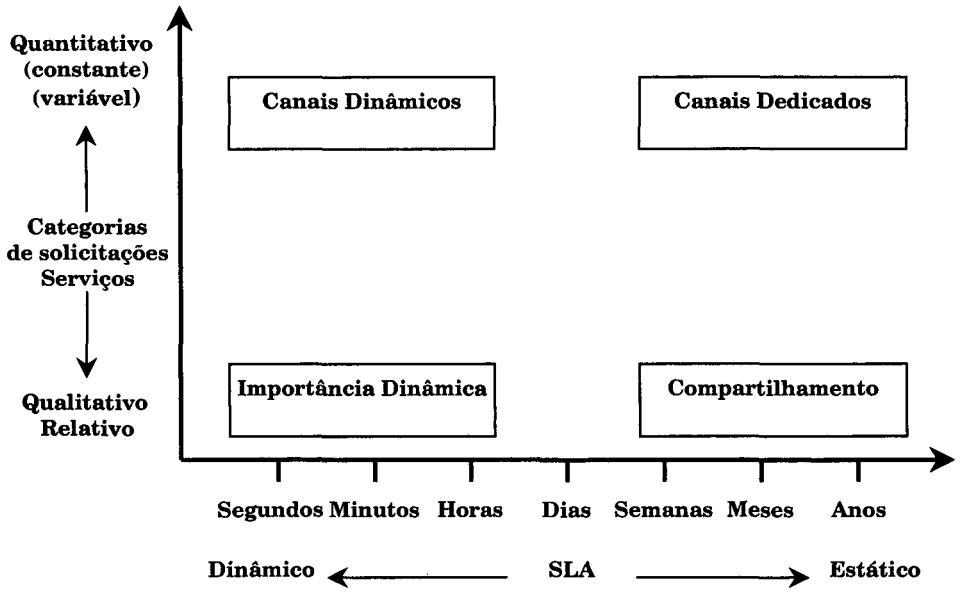

Figura 3.1 Quatro modelos possíveis para a construção de acordos de nível de serviço, ou SLA.

Neste contexto, o modelo de serviço de *canais dinâmicos* refere-se a conexões dedicadas instantâneas, possuindo um forte caráter quantitativo e dinâmico ao mesmo tempo. O modelo de *canais dedicados*, por sua vez, refere-se a conexões dedicadas permanentes, possui forte caráter quantitativo e menos variável no tempo. Sendo que o modelo de serviço de *importância dinâmica* refere-se a conexões compartilhadas instantâneas, possui forte caráter qualitativo e dinâmico ao mesmo tempo. Por último, o modelo de serviço por compartilhamento apresenta forte caráter relativo e mais permanente.

3.1.1.1 Serviço de Canais Dinâmicos

O modelo de *canais dinâmicos* refere-se a solicitação dinâmica por largura de banda exclusiva, feitas pelo cliente, ao provedor de serviços. Este modelo situa-se em um ponto crítico do gráfico, pois a qualquer momento o cliente apresenta a rede uma demanda por uma largura de banda exclusiva, levando o provedor de serviços a dispor de mecanismos de provisão de recursos para decidir se aceita ou rejeita a conexão, como ilustrado pela Figura 3.2.

Neste modelo, assume-se que o usuário final irá fazer uma escolha por um serviço de rede que irá adequar-se aos requisitos da aplicação que pretende utilizar. No entanto, não existe garantia de que isto irá acontecer. Por exemplo, se o usuário requisitar uma conexão de 500Kbps para uma chamada telefônica sobre IP a qual realmente requer somente 20Kbps, então o usuário estará pagando por recursos extras sem que seja notificado pela rede.

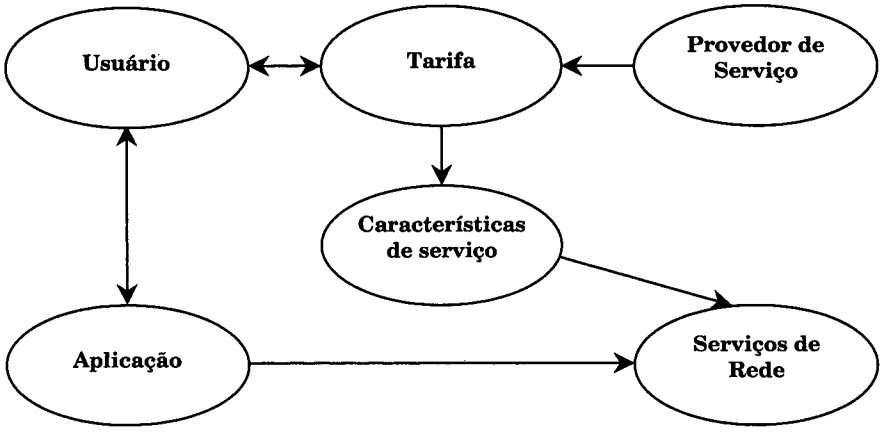

Figura 3.2 Relações entre cliente e provedor de serviço como base no modelo de serviço de canais dinâmicos.

Ainda que parte destas ações possa ser automatizada, o processo básico permanece o mesmo: O provedor de serviços vende serviços de rede baseado em conexões individuais e o cliente toma a decisão de compra baseando-se no preço e no tipo de serviço oferecido. Do ponto de vista dos serviços de rede, em essência, isto significa que o usuário adquire uma capacidade fixa para um destino fixo e fica a cargo da aplicação explorar esta capacidade da melhor maneira possível.

3.1.1.2 Serviço de Canais Dedicados

Diferente do modelo de canais dinâmicos, a natureza dinâmica deste modelo é mais permanente e menos orientada a um único usuário final. Este modelo tem como cliente uma organização, a qual poderá possuir um grande número de usuários finais. Neste contexto, a maior diferença entre estes modelos reside na natureza dinâmica das solicitações por serviço: canais dedicados são mais permanentes, enquanto que canais dinâmicos são estabelecidos em segundos. Uma melhor analogia poderia ser feita tomando-se por base que canais dinâmicos são na verdade conexões dedicadas comutáveis enquanto que canais dedicados são conexões dedicadas permanentes, ambas independentes do propósito de sua utilização.

A Figura 3.3 ilustra um conjunto de relações possíveis entre cliente e provedor de serviços para o modelo de canais dedicados. Neste caso, uma organização adquire um conjunto de canais dedicados entre suas localidades e um número de usuários, desta organização, utiliza os recursos de rede disponíveis. Geralmente, cada usuário inicia uma aplicação, a qual por sua vez utiliza os recursos comuns destinados a organização da melhor maneira possível.

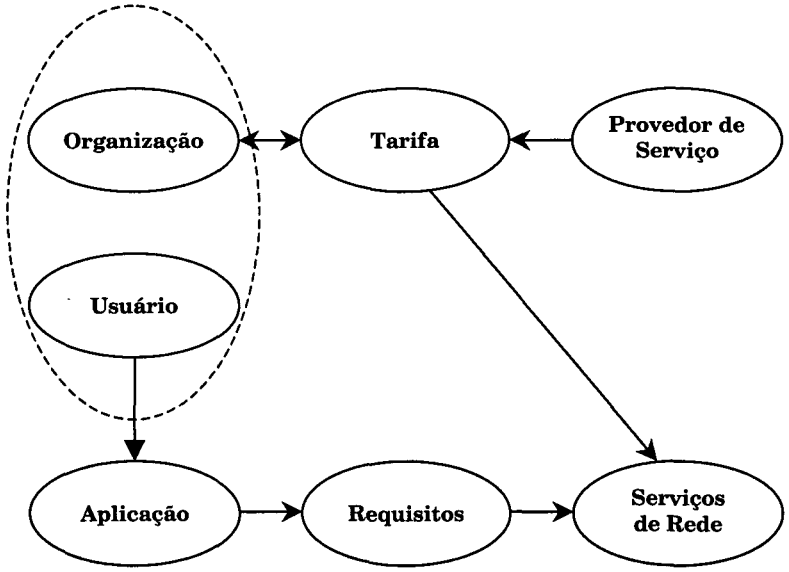

Figura 3.3 Relações entre cliente e provedor de serviço como base no modelo de serviço de canais dedicados.

Nesta configuração, assume-se que não ocorre congestionamento na rede do provedor de serviço. Um mecanismo de controle de congestionamento poderia ser empregado internamente a rede da organização, devidamente na interface entre a rede da organização e a rede pública.

Este modelo é o que mais apresenta demanda, principalmente utilizado para redes *Frame Relay*. Ainda, poderia ser eficientemente implementado, dado que os níveis de carga que se apresentam à rede sejam estáveis o suficiente e que os usuários ativos não apresentem um comportamento de utilização em rajada e sim o mais uniforme possível. Mas, este tipo de comportamento não pode ser obtido sem que se utilize mecanismos de controle de tráfego emitidos pelo usuário.

Ambos os modelos, o de canais dinâmicos e o de canais dedicados, apresentam características que induzem a reserva de recursos por parte do provedor de serviços. Ainda que reserva de recursos possa ser considerado como um princípio contrário aos fundamentos do modelo de Serviços Diferenciados, muitas vezes faz-se necessário que algum tipo de reserva seja proporcionado internamente a este modelo. Ou seja, nada impede que uma determinada marcação de pacotes seja em ultima instancia mapeada para um mecanismo que seja baseado em reserva de recursos.

Este modelo não se adapta perfeitamente quando os requisitos da aplicação do usuário mudam abruptamente. Em particular, serviços de tempo real muitas vezes tornam-se necessários para tornar o modelo em geral atrativo ao cliente. Então, uma questão que permanece em aberto neste modelo reside em solucionar solicitações de tempo real, ou qualquer solicitação em especial,

que possa ocorrer repentinamente. Se não houver um incentivo, por parte do provedor de serviço, de maneira que solicitações especiais devam ser feitas somente quando for necessário então todos os clientes naturalmente irão querer solicitar tratamento especial.

3.1.1.3 Serviço de Compartilhamento de Recursos

Este modelo quando comparado ao modelo de canais dedicados diferencia-se pela adoção da marcação de pacotes ao invés de promover reservas de recursos reais. Este modelo torna possível o melhoramento da multiplexação estatística, mas necessita de mecanismos para resolução de conflitos.

A Figura 3.4 apresenta um conjunto de interações possíveis entre o cliente e o provedor de serviço para o modelo de serviço de compartilhamento de recursos. Neste modelo, clientes adquirem uma cota de compartilhamento de recursos de maneira permanente. Aplicações do cliente exploram a disponibilidade de largura de banda da melhor maneira possível. Em princípio, não existe uma ligação direta entre a aplicação e os serviços de rede, relativos aos pacotes emitidos pela aplicação.

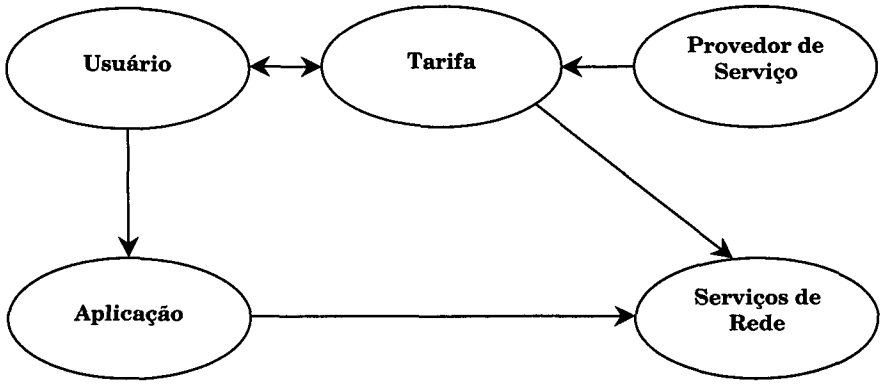

Figura 3.4 Relações entre cliente e provedor de serviço como base no modelo de serviço de compartilhamento.

A maior vantagem do modelo de compartilhamento de recursos reside no fato de que proporciona um serviço simples e consistente. Cada usuário recebe a cota de compartilhamento pela qual pagou, dado que o mecanismo que implemente a partilha seja imparcial o suficiente. Ainda, este modelo proporciona um ambiente favorável a aplicações adaptativas, pois poderá ocorrer diferença significativa sobre a partilha ao longo do tempo e em função do destino.

Um ponto fraco deste modelo aponta para o fato de que o cliente não tem como verificar se realmente está recebendo a cota de compartilhamento pela qual pagou. Então é provável que o SLA será baseado em parâmetros que necessariamente não corresponda à estrutura interna dos serviços de rede. O

operador provavelmente irá aplicar ao modelo de serviço parâmetros qualitativos de desempenho enquanto que internamente a rede, o modelo de serviço será baseado puramente por parâmetros relativos de qualidade.

3.1.1.4 Serviço de Importância Dinâmica

O modelo simples de compartilhamento de recursos, descrito anteriormente, proporciona uma gama de serviços aceitáveis para a maioria dos usuários e aplicações. Algumas necessidades não podem ser satisfeitas pelo modelo de compartilhamento devido a simplicidade implícita em seu nível mais básico, ou seja, qualquer um dos três maiores aspectos de qualidade, *atraso*, *importância* e *taxa de bit* são considerados de propósito específico e não são tratados com a devida importância neste modelo.

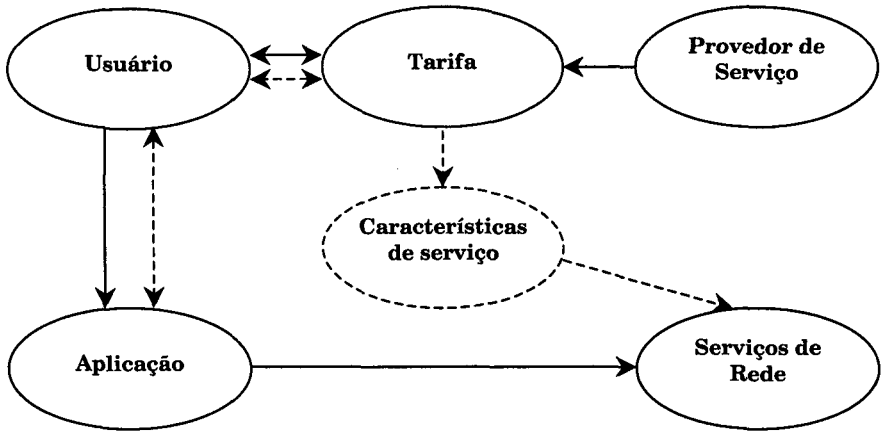

Figura 3.5 Relações entre cliente e provedor de serviço como base no modelo de serviço de importância dinâmica.

O componente chave do modelo de importância dinâmica baseia-se em um mecanismo que informa a rede que um determinado fluxo de tráfego necessita um tratamento melhor do aquele dado a outros fluxos de tráfego. Este mecanismo está representado na Figura 3.5, nas linhas tracejadas.

Este modelo assemelha-se ao modelo de canais dinâmicos, sendo que, neste caso, a principal diferença baseia-se no fato de que os recursos são compartilhados ao invés de haver quaisquer reservas de recursos como ocorre para o caso do modelo de canais dinâmicos.

Pode-se pensar este modelo como um modelo de canais dinâmicos compartilhados, dado que exista um mecanismo de rede que possibilite o gerenciamento da partilha. Neste caso a natureza dinâmica das requisições de serviço está baseada na marcação de pacotes ao invés de ser baseada na reserva antecipada de recurso.

Deve ser observado que este modelo está em conformidade com a proposta do modelo de Serviços Diferenciados. A maior dificuldade imposta neste modelo está na determinação do preço da qualidade extra atribuída ao serviço. Basicamente existem duas opções possíveis: a determinação de um preço em função direta a qualidade requisitada ou um sistema de compensação entre a utilização de serviços normais e serviços especiais da rede por um preço único mensal.

3.2 Classes Abstratas de Serviços de Rede

A diretrizes para especificação de PHBs, como delineadas nos documentos base do modelo de Serviços Diferenciados, preocupa-se em encapsular o nível de abstração de serviços de rede percebida pelo cliente para um nível de abstração de serviços internos a rede. A especificação de uma classe de serviço de rede na sua forma abstrata, de um PHB, permite uma independência dos mecanismos de implementação do comportamento previsto para esta classe.

Esta independência quanto ao mecanismo de rede permite que uma determinada marcação de pacotes, prevista em um PHB, possa ser mapeada para quaisquer mecanismos que possam existir em um ou mais de um dos modelos de serviços de canais dinâmicos, canais dedicados, importância dinâmica e compartilhamento.

Talvez seja esta a maior contribuição do Modelo de Serviços Diferenciados: desvincular o valor semântico da marcação ao longo do caminho do tráfego permitindo que o seu significado seja interpretado no escopo local de um domínio de rede, diminuindo desta forma o grau de complexidade inerente à manutenção de estado ao longo do percurso do tráfego.

3.2.1 Class Selector PHB

De acordo com a RFC 2474 o objetivo da utilização do *Class Selector PHB* é o de manter compatibilidade com a utilização atual dos bits 0-2 do octeto TOS do protocolo IPv4. De acordo com a especificação original do protocolo IP, em momentos de congestionamento somente pacotes que apresentarem um certo nível de precedência é que podem ser admitidos na rede. Devido a utilização destes bits ser similar a utilização previstas para PHBs, pareceu razoável adaptar a utilização corrente destes bits ao modelo de Serviços Diferenciados.

3.2.1.1 Descrição

A RFC 2474 estabelece que um Class Selector PHB deverá proporcionar aos pacotes uma probabilidade de encaminhamento no tempo, que não seja inferior aquela dada aos pacotes marcados para classes inferiores do Class Selector PHB, sob condições normais de operação e de carga na rede.

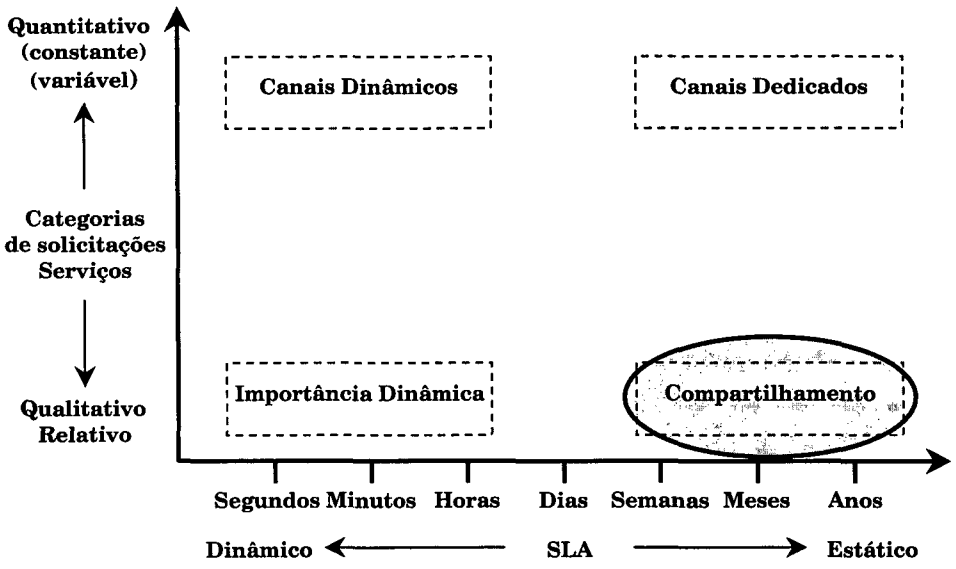

Figura 3.6 Localização do Class Selector PHB entre modelos de serviços.

A codificação do campo DSCP, para este PHB, assume que será utilizado o conjunto de bits 'xxx000', sendo 'xxx' os bits no intervalo de 0-2 passíveis de utilização para efetuar a marcação de fluxos de pacotes para este PHB, de acordo com níveis de precedência entre estes fluxos, não importando a marcação dos bits no intervalo de 3-5.

3.2.1.2 Modelo de Serviço

Este PHB pode ser empregado para os nodos da rede onde se exige uma compatibilidade mínima com o modelo de Serviços Diferenciados, denominados de *Legacy Nodes*. Seu posicionamento entre modelos de serviços é dado na Figura 3.6.

3.2.2 Encaminhamento Expresso

O comportamento definido na especificação da RCF 2598 [Jacobson, Nichols, Poduri 1998], para o PHB de encaminhamento expresso, ou *Expediting Forwarding PHB* (EF PHB), possui um objetivo bem definido e similar ao dos modelos de serviços com características dedicadas. O maior mérito que possa

ser atribuído ao EF PHB é a sua adaptação do modelo de serviços dedicados para o contexto do modelo de Serviços Diferenciados.

3.2.2.1 Descrição

O objetivo do EF PHB é o de encapsular o comportamento de uma classe de serviço que exiba características de perda mínima de pacotes, baixa latência, mínima variação de atraso, largura de banda assegurada e possibilite a aplicação de serviços fim-a-fim através de domínios de rede compatíveis ao modelo de Serviços Diferenciados. A essência de um EF PHB pode ser percebida no próprio requisito imposto ao tratamento de fluxos selecionados para este PHB:

- A taxa de partida de um pacote deve ser igual ou exceder a taxa configurável. Tráfegos EF devem apresentar uma taxa média de transmissão sendo no mínimo igual à taxa configurada, quando medida sobre qualquer intervalo de tempo, sendo este último igual ou não maior do que o tempo de transmissão do pacote para a taxa configurada.

A marcação sugerida pela RFC 2598 de pacotes que devam ser mapeados para este PHB é '101110', sendo que esta classe de serviço possui somente um único nível de importância. Em conseqüência disto, os pacotes que chegam antes do seu tempo de escalonamento impõem três alternativas possíveis aos nodos de borda e interiores: encaminhar o pacote imediatamente; encaminhar o pacote no tempo de escalonamento adequado; ou simplesmente descartar o pacote.

3.2.2.2 Modelo de Serviço

As características intrínsecas ao comportamento previsto para este PHB o elevam para próximo dos modelos de serviços que apresentam requisitos de canais dedicados.

Vale lembrar que classes de serviços que apresentem características de canais dedicados, quer sejam dinâmicos ou permanentes, também podem ser obtidas através da aplicação de algoritmos de multiplexação estatística que apresente como propriedade fundamental uma garantia forte quanto ao isolamento do comportamento de outras classes de serviço. Desta forma a localização desta classe de serviço entre modelos de serviços, como colocado na Figura 3.7, procura evidenciar as exigências inerentes desta classe. Um cenário possível seria aquele que prevê uma infraestrutura de rede que permite a coexistência dos modelos IP e ATM, sendo então redirecionados os pacotes IP com a marcação desta classe de serviço para um circuito virtual garantido da subestrutura ATM.

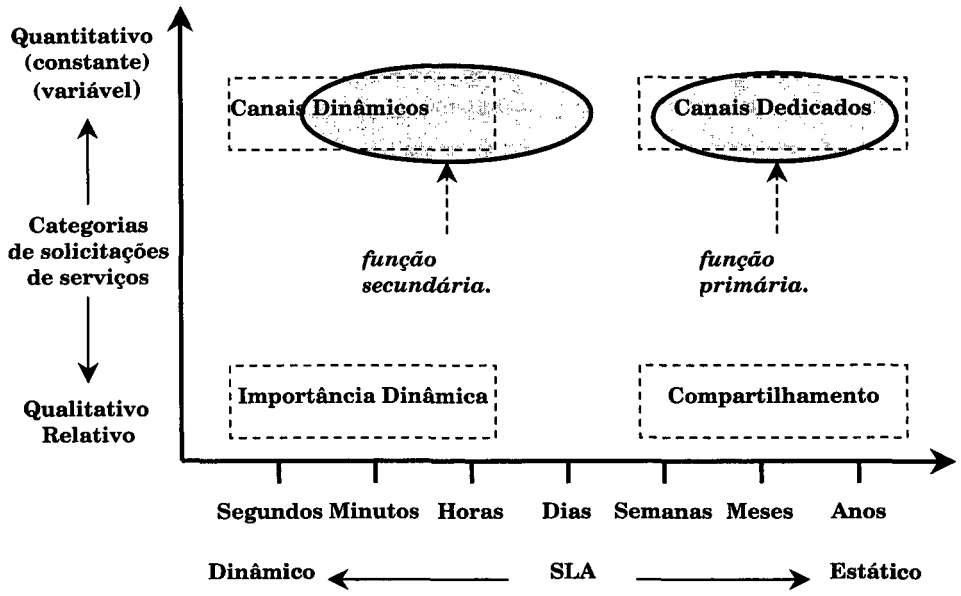

Figura 3.7 Localização do EF PHB entre modelos de serviços.

Como declarado na RFC 2598 em sua introdução, esta classe de serviço, deverá apresentar um comportamento que da perspectiva dos nodos da rede este se pareça com uma conexão ponto-a-ponto ou um circuito virtual dedicado.

3.2.3 Encaminhamento Assegurado

O grupo de PHBs de encaminhamento assegurado, ou *Assured Forwarding PHB* (AF PHB) especificado na RFC 2597 [Heinanen et al, 1999], sugere a criação de uma hierarquia de classes de serviço. Cada classe deverá exibir um determinado comportamento quanto ao método de encaminhamento, independente do comportamento de outras classes, sendo que cada uma deverá apresentar três níveis de precedência quanto à decisão de descarte de pacotes, caracterizada em alta, média e baixa precedência.

3.2.3.1 Descrição

Em princípio um grupo AF PHB pode conter um número N de classes PHBs, cada qual com M níveis de precedência de descarte. A especificação corrente atribui $N = 4$ e $M = 3$, sendo possível a especificação de mais classes ou níveis para uso local [Baker et al. 1999].

Um pacote IP que pertença a uma classe i e possui o nível de precedência de descarte j recebe a marcação no campo DSCP na forma AFij, onde $1 <= i <= N$ e $1 <= j <= M$.

A codificação do campo DSCP sugerida na RFC 2597 para as quatro classes de serviço presentes na especificação do grupo AF PHB é dado na Tabela 3.1.

Precedências de Descarte	*Classe 1*	*Classe 2*	*Classe 3*	*Classe 4*
Baixa	001010	010010	011010	100010
Média	001100	010100	011100	100100
Alta	001110	010110	011110	100110

Tabela 3.1 Codificação do campo DSCP de cada PHB presente na especificação do grupo AF PHB.

3.2.3.2 Modelo de Serviço

Este grupo de PHBs, captura a essência do modelo de Serviços Diferenciados, sugerindo a diferenciação de serviços de rede através da diferenciação do método de encaminhamento dos pacotes pertencentes a cada classe.

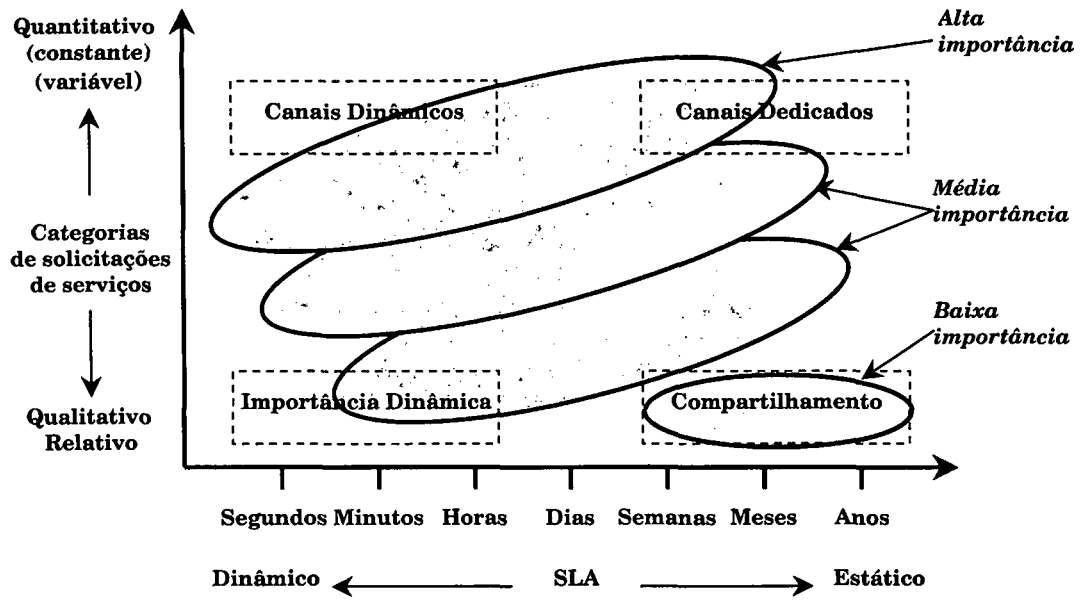

Figura 3.8 Localização do AF PHB entre modelos de serviços

Cada classe deverá exibir um nível de importância em relação a outra classe. Como cada PHB deve exibir um comportamento distinto em termos de características de atraso, variação de atraso, perda de pacotes e prioridade

relativa entre PHBs, estes níveis de importância serão obtidos em função das restrições impostas a estas características. Desta forma, a localização das quatro classes de serviço previstas neste PHB poderiam assumir posições como ilustrado na Figura 3.8. Onde restrições mais rígidas elevam uma classe de serviço para as características de serviços dedicados. Enquanto que restrições mais brandas aproximam ao comportamento de serviços compartilhados.

3.3 Mecanismos de Diferenciação de Serviços

Mecanismos representam a realização dos níveis de abstração de classes de serviços ao cliente e seu mapeamento correspondente a classes abstratas de serviços de rede. Mecanismos podem ser agrupados em duas categorias: mecanismos de condicionamento de tráfego e mecanismos de diferenciação de serviços. A primeira ocupa-se em policiar os fluxos de tráfego de forma a adequá-los ao comportamento da classe. A segunda se ocupa em proporcionar uma plataforma sobre a qual os serviços de rede possam ser realizados e determinam as regras de convivência entre diversas classes de serviços passíveis de serem aplicadas em um mesmo ponto da rede.

Dada a relevância deste tópico, descrições mais detalhadas serão desenvolvidas no capítulo 4. Esta seção somente demonstra que mecanismos de diferenciação de serviços constituem na verdade classes de serviços de rede na sua forma realizável.

3.4 Resumo

Este capítulo procurou evidenciar a importância da diferenciação dos serviços de rede no nível de abstração do cliente, sendo que este pode assumir o papel de um indivíduo, uma organização ou uma aplicação. Em função disto procurou-se distinguir formas de solicitações por serviços de rede por seus caracteres quantitativo, qualitativo e relativo. Estas formas de solicitações quando dispostas ao longo do tempo determinam a natureza dinâmica das imposições de requisitos sobre a rede. Esta percepção permite ao provedor de serviços localizar modelos de serviços que atendam as características implícitas destas formas de solicitações.

Na seção 3.2, Classes Abstratas de Serviços de Rede, procurou-se evidenciar que o modelo de Serviços Diferenciados determina diretrizes para especificação de classes de serviços de rede, ou PHBs, especificados em um

nível de abstração elevado quanto ao comportamento esperado de uma classe de serviço de rede.

Portanto, uma investigação mais adequada faz-se necessário quanto a realização destes níveis de abstração na forma de mecanismos de rede. Pois nenhuma diferenciação de serviços será obtida em nenhum dos níveis de abstrações mais elevados se não houver mecanismos que possam garantir a realização destes níveis de abstração na forma de serviços de redes efetivos.

Capítulo 4

Mecanismos de Controle de Tráfego e de Diferenciação de Serviços

Todo o modelo de Serviços Diferenciados demonstra uma preocupação extrema em definir um arcabouço que realce a descrição do comportamento esperado para uma classe de serviço de rede, de maneira que esta seja independente do mecanismo que a implemente. No entanto, o projeto destas classes de serviço na forma de PHBs, quando especificados, fazem uma referência implícita, ou um direcionamento não intencional, a determinados mecanismos para a sua implementação. Como ocorre na seção 2.4 da RFC 2474, onde faz-se referência a um PHB como um elemento divisor da largura de banda. Neste caso, a referência implícita recai diretamente para aqueles mecanismos que proporcionam o compartilhamento hierárquico de banda. Onde é estabelecido que um determinado PHB receba no mínimo a sua cota de compartilhamento e que a largura de banda excedente seja distribuída entre os PHBs concorrentes.

Nenhuma classe de serviço irá demonstrar o comportamento esperado se não houver um controle de admissão do tráfego sobre estas classes de serviço. Desta forma o termo mecanismo pode ser dividido em duas categorias: uma utilizada para o condicionamento do tráfego e outra para implementar a visão abstrata especificada em um PHB ou classes de PHBs. Este capítulo apresenta duas seções relativas a estas categorias de mecanismos.

4.1 Mecanismos de controle de tráfego

O processo de controle de tráfego normalmente é aplicado aos nodos de borda de um domínio de rede compatível ao modelo de Serviços Diferenciados. Neste ponto, deverão ocorrer as ações de classificação, medição, marcação, conformação do tráfego e aplicação de políticas de descarte de pacotes. Estas ações, no entanto, estão relacionadas aos modelos de serviços adotados, os quais impõem restrições aos fluxos de tráfego que apresentam-se a rede.

Em uma rede genuinamente compatível ao modelo DS, presume-se que também o cliente segue as diretrizes deste modelo. Desta forma, ações de classificação, conformação de tráfego e de descarte poderiam ser tomados no lado do cliente. Neste cenário, restaria poucas ações a serem tomadas internamente a rede do provedor de serviços.

O processo de marcação sugere ações relativas a codificação do campo DSCP do protocolo IP a marca correspondente ao PHB selecionado. No entanto, a maior função deste mecanismo é o de selecionar níveis de importâncias de um determinado pacote dado à taxa de ocupação do canal pela classe de serviço que este pacote pertença. Onde nível de importância, neste caso, não faz referência aos níveis de importância relativos entre classes de serviços e sim quanto à disponibilidade de recursos.

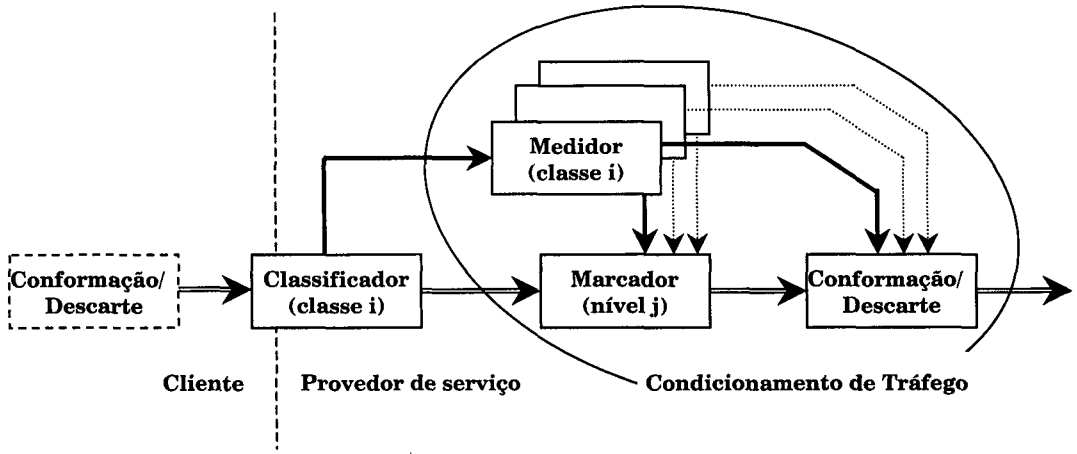

Figura 4.1 Mecanismos de controle de tráfego aplicado ao nodos de borda de um domínio de rede compatível ao modelo de Serviços Diferenciados.

Como ilustrado pela Figura 4.1, o medidor é um mecanismo aplicado para a determinação da taxa de ocupação instantânea de cada agregado atribuído a um determinado PHB. O resultado destas medições influencia diretamente o processo de marcação do nível de importância de um pacote que chega para esta classe, o que pode leva-lo a ser aceito para transmissão, rebaixado para uma classe inferior ou marcado simplesmente para o descarte imediato.

O processo de conformação do aporte de tráfego sobre a rede tenta adequar o comportamento bruto de um determinado tipo de tráfego ao perfil exigido pela classe de serviço. Desta forma, um tráfego que esteja em conformidade ao perfil sofrerá ações mínimas ou mesmo nenhuma de descarte.

O processo de classificação, em uma rede genuinamente compatível ao modelo DS teria ações mínimas se os pacotes ao apresentarem-se a rede possuíssem uma codificação, em seus campos DSCP, compatível a classe selecionada. Mas isto dependerá da visão do cliente e do provedor de serviços, sobre as classes de serviços disponíveis e da interpretação feita pelo cliente da classe selecionada.

4.1.1 Classificação

Um *classificador* é um mecanismo utilizado para selecionar uma classe PHB em função do comportamento exibido por um determinado fluxo, o qual deve ser expresso por uma marca. Vale lembrar que o comportamento de um fluxo é determinado com base nas características intrínsecas de vazão, atraso, variação do atraso, perda e prioridade relativa. Portanto, a semântica atribuída a uma marca deverá capturar o comportamento de fluxos distintos permitindo a diferenciação de um fluxo de outro. Ainda, pode ocorrer que diversos fluxos de trafego exibam um comportamento que os remetam a uma mesma marca, constituindo desta forma um agregado.

Ainda que pareça uma tarefa puramente técnica, alguns problemas de escala podem ocorrer se um número muito grande de requisitos for levado em consideração na decisão de agregação. Devido à tomada de requisitos depender do modelo de serviço adotado, pelo provedor de serviços, este irá influenciar diretamente nos níveis de agregação atribuídos a uma classe de serviço ou PHB. Desta forma, pode-se conjecturar sobre os seguintes modelos possíveis de classificação:

1. O usuário seleciona uma classe de serviço específica entre uma gama de classes de serviços disponíveis.

2. A aplicação seleciona automaticamente uma classe se serviço em função das características do fluxo originado por ela.

3. A rede seleciona uma classe de serviço apropriada com base na informação produzida por uma aplicação.

4. A rede seleciona uma classe de serviço apropriada com base em contratos com o usuário não importando os requisitos da aplicação.

5. Uma combinação possível dos métodos anteriores.

O primeiro método exige que informações de classificação sejam transmitidas entre o cliente e a rede. Se o cliente não estiver satisfeito com a classificação obtida então um mecanismo para sinalizar o grau de satisfação do cliente é necessário. Ainda, nas diretrizes colocadas pelos documentos base do modelo de Serviços Diferenciados, informações sobre fluxos individuais e sobre o cliente não deverão ser utilizadas nos nodos que compõe o núcleo da rede.

O segundo método mostra se o mais adequado para redes que genuinamente adotem o modelo de Serviços Diferenciados. Esta alternativa pressupõe que aplicações do usuário e a rede atribuam o mesmo valor semântico para a marca. Neste cenário, o problema da classificação estaria resolvido. Mas, também, as diretrizes bases do modelo DS determinam que este modelo deverá ser independente da aplicação e que deverá ser possível de ser empregado utilizando-se aplicações e mecanismos de redes correntes, os quais ainda não apresentam este grau de refinamento.

A questão imposta pelo segundo método exige a existência de um mecanismo de marcação, o qual deverá atribuir um valor apropriado ao campo DSCP de cada pacote, no equipamento do cliente. Se este mecanismo não estiver disponível, então uma solução que se apresenta razoável é a de classificar pacotes em nodos de borda, primeiramente com base em outros campos do pacote, para então receberem uma marca correspondente no campo DSCP. O que caracteriza o terceiro método.

Como sugere o quarto método, o cliente somente tem acesso à classe de serviço pré-definida em termos contratuais. Neste caso, todos os pacotes que tiverem como origem um cliente em específico será mapeado para o PHB apropriado.

O último caso, sugere por si mesmo, uma combinação de todos os métodos. Neste caso, uma boa combinação seria fazer com que o nodo de borda decida sobre a classe de serviço, baseando-se na identificação produzida pela aplicação. Se houver uma marcação adequada no campo DSCP destas aplicações, então nenhuma ação de modificação será necessária. Caso contrário, a decisão de classificação irá basear-se em outras fontes de informação produzidas pela aplicação tais como: endereço de origem e de destino, porta de origem e de destino e identificação de protocolo.

4.1.2 Medição

O propósito principal da medição no contexto do modelo de Serviços Diferenciados é o de ordenar pacotes classificados, em seus diversos níveis de

importância[7]. A lógica de condicionamento de tráfego prevista para este modelo assume que as ações de marcação, conformação e descarte, estão baseadas no resultado da medição do comportamento de uma classe[8], em um dado instante, de maneira a atribuir um nível de importância aos pacotes desta classe.

4.1.2.1 Objetivo da Medição

O objetivo da medição, colocado por [Kilkki, 1999], deve levar em consideração quatro aspectos bases: a *versatilidade*, *custo*, *imparcialidade* e *robusteza*. Quanto a versatilidade, um método de medição deverá funcionar propriamente para a maioria das situações possíveis, mesmo para aquelas onde ocorrem a presença de padrões de tráfegos "estranhos". O aspecto de custo se refere a complexidade do método de mensuração, onde um modelo teórico perfeito muitas vezes são impraticáveis e podem ser substituídos por soluções simples e eficientes. O aspecto de imparcialidade se refere a consistência das regras de diferenciação que possam ser aplicadas, por exemplo, dada a quádrupla {PHB, nível de importância, intervalo de tempo, capacidade} o custo do pacote deveria ser aproximadamente o mesmo.

Portanto, uma interpretação possível para imparcialidade poderia ser a seguinte:

> *Em um compartilhamento imparcial, dada a quádrupla {PHB, nível de importância, intervalo de tempo, capacidade}, o custo de um pacote deverá refletir apropriadamente o modelo de serviço aplicado pelo provedor de serviço* [Kilkki, 1999].

Como ilustrado pela Figura 4.2, onde está caracterizado o modelo de serviço garantido e o modelo de serviço de compartilhamento de recursos, o termo medição pode acarretar objetivos distintos. No primeiro modelo o usuário obtém uma capacidade fixa, sendo então o objetivo da medição certificar se este limite foi ultrapassado. Para o segundo modelo, medir implica em determinar a taxa de bit instantânea ocupada pela classe de serviço, de maneira a averiguar o comportamento da classe em um determinado instante.

---

[7] Níveis de importância, neste caso, possui a conotação de que um pacote possa ser marcado para ser transmitido ou para ser descartado, em função da capacidade disponível de recurso em um dado instante.

[8] Neste caso, novamente, faz-se referência a uma classe ou PHB na forma expressa de um agregado, portanto passível de ser medida em termos de ocupação do canal.

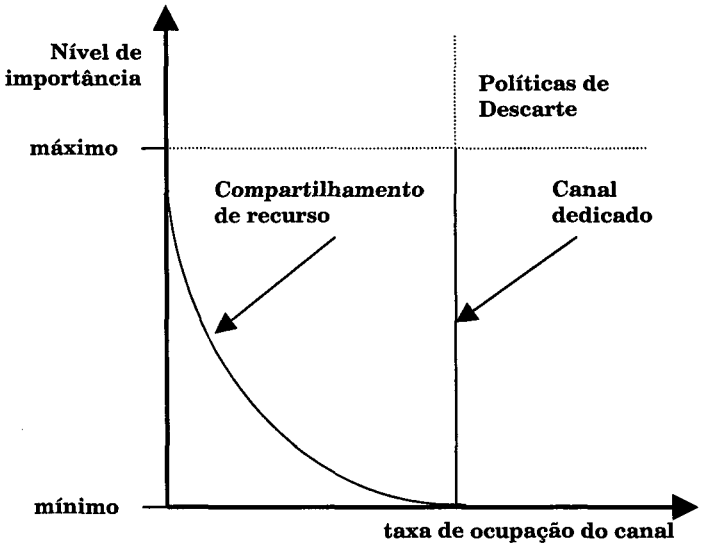

Figura 4.2 Princípios atribuídos a medição para os modelos: serviços garantidos e compartilhamento de recurso.

Portanto, assumindo-se que o nível de importância siga como a curva indicada na Figura 4.2 para o modelo de serviços compartilhados, a medida para a qual a capacidade de banda disponível a uma classe de serviço tenda para o seu limite também o nível de importância dos pacotes desta classe tenderão para um valor mínimo de importância. No limite, serão adotadas políticas de descarte ou de rebaixamento de um pacote para uma classe de serviço inferior.

4.1.2.2 Mecanismos de Medição

Entre os principais mecanismos de medição encontram-se o token bucket e suas variantes, bem como, a aplicação do método de obtenção de uma média exponencial móvel, ou EWMA.

4.1.2.3 Token Bucket

O princípio base de um *token bucket*[9] presume a existência de um recipiente de capacidade *b*, o qual será esvaziado a uma taxa constante *r*. O resultado da formulação determina quando o recipiente estará vazio. Uma descrição formal de um *leaky bucket* pode ser encontrada em [Parekh e Gallager 1993].

Em um mecanismo do tipo token bucket, fichas ou créditos conferem a permissão ao aporte de uma determinada quantidade de dados sobre a rede. A

[9] O termo token bucket faz referencia a um recipiente na forma de funil o qual contem fichas ou créditos. Também denominado de *leaky bucket* ou gotejador.

taxa para a qual as fichas são geradas determina a taxa média de longo termo permitida pelo mecanismo de token. Um mecanismo de token deste tipo é denominado de token estrito, o qual concede o direito de passagem a um pacote somente quando houver uma ficha disponível.

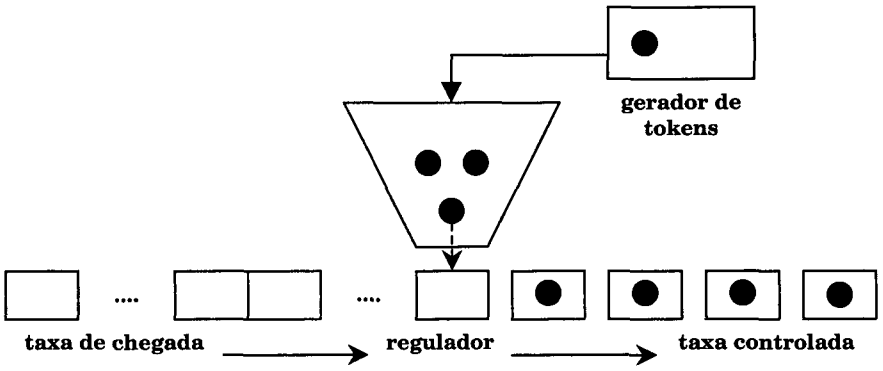

Figura 4.3 O mecanismo token bucket como um elemento de medição da taxa de ocupação do canal por uma classe de serviço.

Uma variação deste modelo, ilustrada pela Figura 4.3, pode ser utilizada para a ação de medição utilizando-se um acumulador de fichas. A profundidade do acumulador determina a capacidade máxima admitida de tráfego proveniente de uma classe de serviço. À medida que fichas são retiradas em função da passagem de tráfego desta classe, pode-se determinar também a sua taxa de ocupação do canal e conseqüentemente proceder a medição.

4.1.2.4 Média Exponencial Móvel

Média exponencial balanceada móvel, ou *Exponencial Weighted Moving Average (EWMA)*, é um mecanismo que baseia-se no conceito de janelas de tempo, *T*, para a qual é permitida a passagem de determinado número, *N*, de pacotes.

O limite do número de pacotes permitidos em cada janela é dinamicamente atualizado em função do valor médio dos *N* pacotes permitidos em uma janela e de uma soma de termos, *S*, os quais levam em consideração o comportamento passado, ou seja, o número de pacotes aceito nos intervalos anteriores. Mais especificamente, a contribuição de cada termo desta soma diminui exponencialmente a medida para a qual as janelas se distanciam no tempo. A regra de calculo do valor de N para a *i*-ésima janela é:

$$N_i = \frac{N - \lambda S_{i-1}}{1-\lambda} \quad para \quad 0 < \lambda < 1 \tag{4.1}$$

onde,

$$N = \lambda T \quad = C\lambda T S_{i-1} \quad = (1-\lambda)x_{i-1} + \lambda . S_{i-2} \tag{4.2}$$

sendo x_{i-1} o número de pacotes aceitos na $(i-1)$-ésima janela.

O parâmetro λ é um fator constante de balanceamento, o qual faz com o mecanismo seja mais ou menos flexível com respeito ao aporte em rajadas. Se $\lambda = 0$, N_i será constante. Um valor de $\lambda > 0$ torna o mecanismo mais tolerante a flutuações estatísticas (rajadas maiores são aceitas), reduzindo desta forma a probabilidade de falsos alarmes. Mas, a medida em que λ aumenta, a capacidade de resposta diminui. Portanto o desempenho do método EWMA depende fortemente dos valores escolhidos para λ: para fontes bem comportadas um valor alto é necessário; para fontes não comportadas este valor tem que ser baixo.

O limite deste mecanismo reside no valor estático de λ, o qual deve ser atribuído a priori. Uma vez que o valor de λ foi escolhido, a probabilidade de falso alarme somente pode ser reduzida aumentando-se o valor de T. Isto também contribui para a redução do tempo de resposta do sistema. O problema pode ser resolvido introduzindo-se o fator de superdimensionamento, $C \geq 1$, o qual em conjunto com o par de valores fixos T e λ diminuem a probabilidade de falso alarmes, [Catania et al. 1996].

4.1.3 Marcação

O processo de marcação tem dois objetivos bases: o primeiro diz respeito a marcação de pacotes acima e abaixo de um fator de limiar da taxa de ocupação do canal (na sua forma mais genérica poderá haver diversos níveis de limiares). O segundo reside em determinar quão acima ou abaixo, de um fator padrão de limiar, o resultado da medição se encontra.

Da perspectiva do modelo de Serviços Diferenciados o objetivo principal em ambos os casos é o de mapear pacotes para um dos níveis de importância disponíveis em uma classe PHB. Desta forma, a marcação real que um pacote irá receber depende do resultado da medição para um PHB em particular, do tráfego total emitido pelo cliente e da carga total da rede do usuário final.

4.1.4 Conformação

O processo de conformação de tráfego procura tratar o comportamento bruto exibido por um fluxo de pacotes de maneira a adequá-lo ao comportamento previsto para o perfil da classe da qual pertença. Ao invés de um fluxo

experimentar rebaixamento, remarcação ou descarte de seus pacotes, melhor será condicionar o tráfego para próximo do perfil da classe.

Conformação de tráfego é efetuada por duas funções predominantes: uma para suavizar o aporte de tráfego, de forma que este seja limitado a uma taxa pré-determinada, e outra, para impor ao tráfego uma taxa média de longo termo e admitir aportes eventuais a taxas maiores.

4.1.4.1 Suavização

Uma função de suavização de tráfego é utilizada como mecanismo que através do qual fluxos de pacotes que apresentam um comportamento de aporte errático, possam ser conformados de maneira que apresentem um comportamento estável para a rede. Este mecanismo pode ser implementando através de um token bucket, cuja profundidade comporte somente a passagem de um único pacote, sendo que e a taxa de geração de novos créditos (fichas) deverá ser configurada para a taxa de suavização requerida.

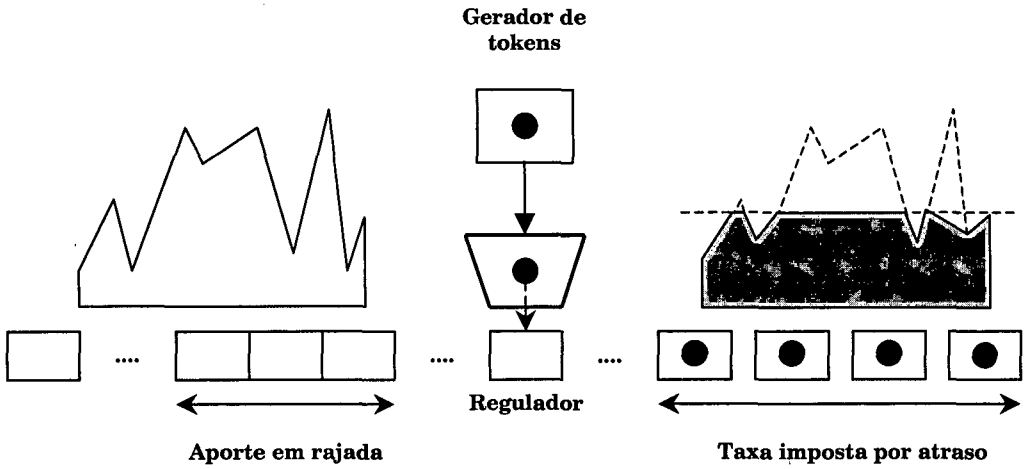

Figura 4.4 Mecanismo de suavização de tráfego. Um token bucket, nesta configuração é denominado de *leaky bucket*, ou gotejador.

4.1.4.2 Conformação do Aporte em Rajada

Outro método utilizado para conformação de tráfego, bem como, para o controle da taxa de ingresso de pacotes na rede refere-se aa conformação do perfil de aporte em rajada. Este método difere do método de suavização devido a profundidade estabelecida para o regulador token bucket. Enquanto que o método de suavização tem como objetivo impor uma taxa controlada de admissão a rede em função da taxa de geração de créditos, o mecanismo de conformação de aporte em rajada acumula créditos para admitir na rede rajadas de perfil controlado. Portanto, quando houver um aporte em rajada apresentando-se a rede, a passagem deste aporte será permitida de

acordo com a quantidade de créditos acumulados pelo token bucket, como ilustra a Figura 4.5.

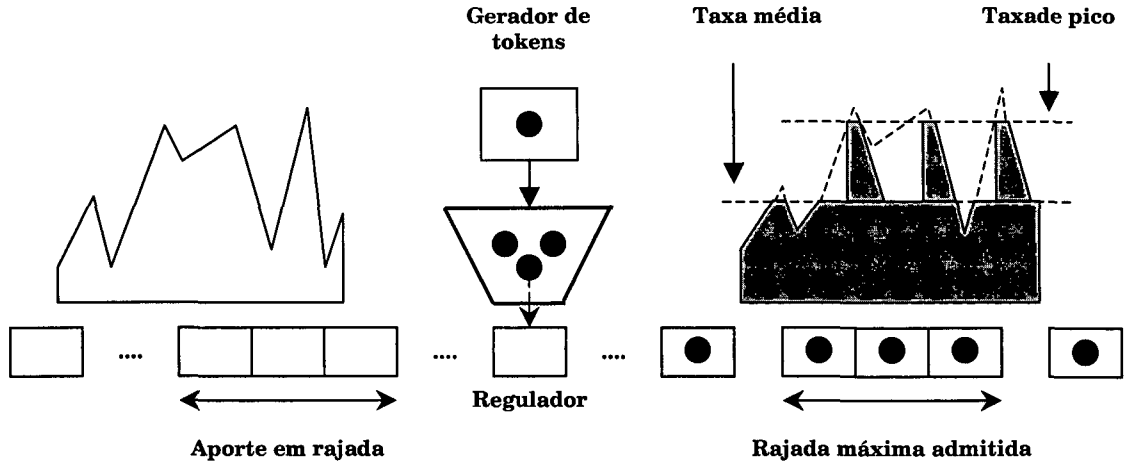

Figura 4.5 Mecanismo de conformação do aporte de rajadas implementado por um token bucket.

Outra configuração possível para proporcionar a conformação de tráfego é a utilização de tokens bucket em série, conjuntamente ao classificador de pacotes. Um destes tokens bucket determina a taxa de aporte médio, enquanto que um segundo determina a taxa de aporte de pico, diferindo-se apenas pela configuração de seus parâmetros. Desta forma, a aplicação em conjunto destes mecanismos pode condicionar o comportamento bruto do tráfego para o perfil esperado da classe.

4.1.5 Descarte

O processo de descarte de pacotes ocorre quando são violadas as regras previstas para o perfil de uma classe. O descarte constitui uma métrica para avaliação do comportamento de uma classe de serviço, bem como, constitui um parâmetro de qualidade de serviço. Nenhuma metodologia de diferenciação de serviços é alcançada se não houver controle de admissão em uma classe. Este parâmetro pode ser utilizado para sinalizar ao cliente sobre o aporte imposto por ele sobre a classe de serviço selecionada, bem como ao provedor de serviço sobre o limite de capacidade alcançada.

Políticas de descarte de pacotes pode determinar o descarte sumário de pacotes, como também o rebaixamento de um determinado pacote para uma classe de serviço inferior.

4.1.5.1 Descarte pela Cauda

O mecanismo default para descarte de pacotes é aquele utilizado por disciplinas de fila do tipo FIFO. Neste mecanismo, o primeiro pacote a ser servido é o primeiro a chegar no sistema. Quando não houver mais espaço na fila, pacotes que chegarem a partir de então assumem a última posição da fila, provocando o descarte do pacote que nesta posição se encontrava. Este mecanismo não exibe uma ação de descarte imparcial, pois fontes de tráfego que apresentam um caráter de aporte em rajada penalizam fontes de tráfego que apresentam comportamentos adequados.

4.1.5.2 Descarte Antecipado

Neste método um pacote pode ser descartado, mesmo que exista recursos suficientes na fila. Geralmente é aplicado quando a fonte de tráfego foi condicionada ao perfil de uma classe de serviço, sendo então o seu objetivo proporcionar uma forma imparcial quanto à decisão de descarte de um pacote de uma destas classes.

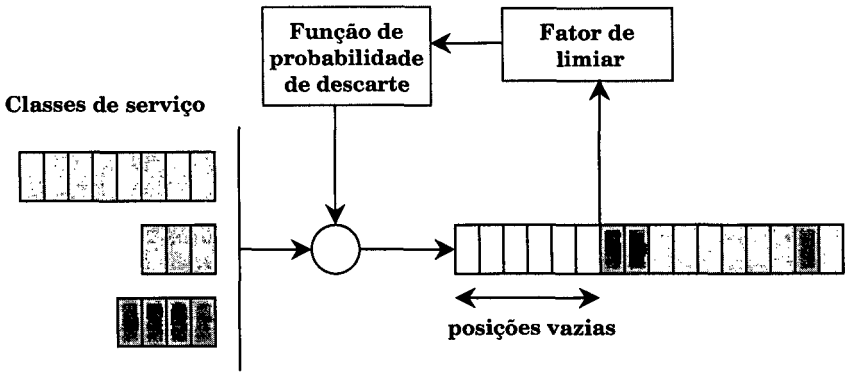

Figura 4.6 Descarte antecipado de pacotes em função do comportamento das classes de serviço.

Este método também permite sinalizar a fonte de tráfego que está contribuindo para a exaustão de recursos, sobre a capacidade disponível, e que a insistência neste tipo de comportamento aumentará a sua probabilidade de descarte.

4.1.5.3 Descarte Randômico

O descarte do pacote que tenha chegado recentemente não se caracteriza como um único método de descarte de pacotes. A saturação do buffer de saída também constitui um método de sinalização para o descarte de pacotes, sendo que a decisão de descarte pode recair sobre qualquer pacote presente no buffer até o momento, de maneira a liberar espaço para o pacote que chega.

Este método pretende implementar uma função de descarte que seja imparcial entre as classes de serviços, pois parte do princípio de que uma fonte de tráfego que esteja contribuindo para a exaustão de recursos também possui uma alta probabilidade em sofrer a ação de descarte.

4.2 Mecanismos de diferenciação de serviços

Mecanismos de diferenciação de serviços, por sua vez, preocupam-se em proporcionar uma arquitetura sobre a qual possa ser realizada a distinção entre classes de serviço. Esta arquitetura é construída com base em dois elementos fundamentais: uma estrutura de dados para armazenamento da informação, normalmente uma fila, e um algoritmo de escalonamento de pacotes, o qual por sua vez é orientado pelas restrições previstas para o comportamento da classe.

O termo disciplina de fila geralmente é utilizado para referir a ambos os aspectos, organização da estrutura de dados e o método de acesso a esta estrutura. Como o processo de escalonamento determina o comportamento da disciplina de fila, serão apresentados a seguir alguns exemplos com base no comportamento do escalonador.

4.2.1 Escalonamento FIFO

O método *First In First Out (FIFO)*, constitui o elemento base para a construção de mecanismos mais sofisticados. Este é expresso por uma fila onde a ordem de atendimento, ou de serviço solicitado ao escalonador, é dada em função da ordem de chegada dos pacotes. Neste caso o primeiro a chegar será o primeiro a ser servido pelo escalonador, como mostra a Figura 4.7.

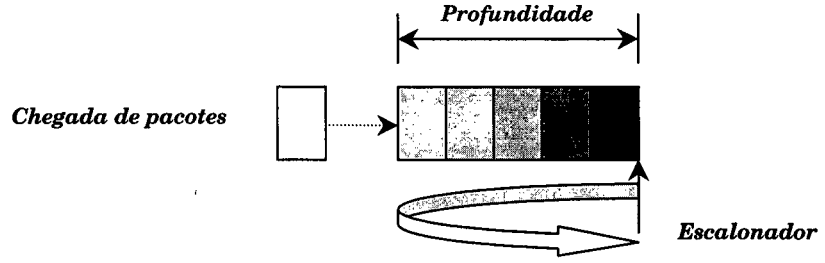

Figura 4.7 Mecanismo FIFO. Elemento fundamental de estruturas de diferenciação de serviços.

Compreender este mecanismo fundamental é essencial para a compreensão de algoritmos de ordem superiores. A profundidade da fila está diretamente relacionada a largura de banda atribuída a classe de serviço. Esta, por sua vez, induz um atraso inerente ao tempo de atendimento de toda fila. Admitindo-se pacotes de mesmo tamanho, o atraso induzido a cada pacote é uma razão direta do tempo de atendimento do número de pacotes presentes em toda a fila. Desta forma, o objetivo deste algoritmo é o de servir tão rápido quanto possível a todos os pacotes presentes na fila, sob pena de estar desperdiçando largura de banda. Como largura de banda, ou taxa de bits, pode ser expressa em tempo de atendimento, este mecanismo constitui um relógio, o qual poderá andar rapidamente ou ser atrasado em função da taxa de chegada de pacotes e das variações possíveis no tamanho de cada pacote. Mecanismos externos, tais como um regulador da taxa de chegada e um relógio externo baseado no tempo de atendimento de um pacote de tamanho médio podem sincronizar o relógio interno deste mecanismo, de maneira a que este se comporte como em um sistema fluido ou síncrono.

Aproximações deste grau de sincronismo é o objetivo implícito as muitas variações de algoritmos propostos para implementar comportamentos característicos exigidos pelas classes de serviço que são sensíveis a requisitos temporais.

4.2.2 Escalonamento Fluido de um Nível (GPS)

Um escalonador fluído é um modelo que tem por base um sistema fluído no qual pressupõe-se a existência de pacotes fluidos, portanto infinitamente divisíveis, os quais concorrem por um único canal de saída e são escalonados de acordo com uma cota de compartilhamento deste canal. Este modelo, geralmente denominado de GPS, ou *Generalized Processor Sharing*, é um modelo base para uma série de algoritmos aproximados. Pois, o objetivo da maioria destes algoritmos é o de aproximar um modelo discreto orientado por pacotes a este modelo síncrono orientado por pacotes fluidos.

Este modelo garante a cada classe de serviço, como ilustrado na Figura 4.8, no mínimo a sua cota de compartilhamento, em momentos de congestionamento, sendo que fora deste intervalo o excesso de banda disponível no canal de saída é distribuído de forma imparcial entre as classes de serviços que apresentarem demanda reprimida de acordo com suas cotas de compartilhamento. Portanto caracteriza um modelo base para uma classe PHB cujo comportamento apresente níveis de prioridade distintos de seus PHBs individuais. A descrição formal do modelo GPS é dada como segue:

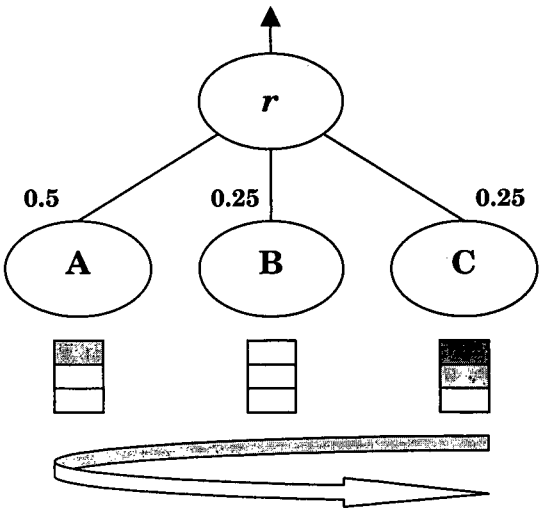

Figura 4.8 Estrutura do escalonador fluido GPS. As classes A e C apresentam demanda reprimida por serem servidas pelo escalonador.

Um escalonador GPS de um nível, o qual contenha N filas é caracterizado por N números reais positivos ϕ_i, onde $i = 1, \ldots, N$, os quais definem as cotas de compartilhamento de cada fluxo. Durante qualquer intervalo de tempo, $\tau = t_2 - t_1$, quando houver exatamente M filas não-vazias, $B(\tau)$, o escalonador irá escalar M pacotes do topo da fila simultaneamente, em proporção a suas cotas de compartilhamento. Seja $W_i(t_1, t_2)$ o fluxo i escalonado sobre o intervalo $(t_1, t_2]$, então um escalonador GPS, em regime conservativo, é definido pela equação:

$$\frac{W_i(t_1,t_2)}{\sum_{j \in \mathbf{B}(\tau)} W_j(t_1,t_2)} \geq \frac{\phi_i}{\sum_{j \in B(\tau)} \phi_j} \quad para \ j \in B(\tau) \tag{4.3}$$

a qual estabelece que a cota de compartilhamento pode ser expressa em função da taxa de ocupação apresentada pelo fluxo i, [Parekh, Gallager, 1993]. A partir desta definição, segue imediatamente que,

$$\frac{W_i(t_1,t_2)}{\phi_i} = \frac{\sum_{j \in B(\tau)} W_j(t_1,t_2)}{\sum_{j \in B(\tau)} \phi_j} \quad para \ j \in B(\tau) \tag{4.4}$$

é satisfeita para quaisquer dois fluxos i e j ativos durante o intervalo $(t_1, t_2]$. Ou seja, o escalonador distribui o recurso de banda de forma imparcial entre todos os fluxos ativos, em proporção a suas cotas de compartilhamento do canal. Pois, se a diferença destas relações não for igual a zero significa que

algum fluxo está recebendo recurso além de sua cota, o que permite estabelecer uma medida de imparcialidade para métodos aproximados.

Para facilidade de discussão e sem perder o caráter genérico, assume-se que aos números ϕ serão atribuídos valores de forma que $\sum_i^N \phi_i = 1$. Portanto, seja $r_i = \phi_i r$ a taxa de serviço obtida pelo fluxo i onde r é a taxa de serviço do escalonador, pode ser mostrado que,

$$g_i = \begin{cases} \dfrac{\phi_i}{\sum_{j \in B(\tau)} \phi_j} r & para\ j \in B(\tau) \\ 0 & para\ outros\ casos \end{cases} \tag{4.5}$$

a qual estabelece que o fluxo i receberá uma taxa de serviço garantida g_i enquanto $r_i \leq g_i$, não importando o comportamento dos outros fluxos no intervalo τ, sendo zerada para outros casos (o que significa conter a classe na sua cota de compartilhamento).

Como este modelo apresenta uma garantia de atraso forte, também pode ser utilizado para o pior caso, com relação ao atraso de um fluxo de tráfego que esteja restringido por um *leaky bucket* que apresente uma taxa não maior do que r_i, como demonstrado em [Parekh, Gallager 1993].

4.2.3 Escalonamento Fluido hierárquico (HGPS)

O escalonador fluido hierárquico, HGPS, apresentado por [Bennet e Zhang 1997], é uma extensão direta do escalonador fluido de um único nível, GPS, para um escalonador hierárquico de diversos níveis. Como em um nodo da rede compatível ao modelo de Serviços Diferenciados, poderão coexistir diversas classes PHBs, cada qual concorrendo pelo recurso de banda de um mesmo canal, faz-se então necessário um algoritmo que proporcione o compartilhamento de recursos entre estas classes, como ilustra a Figura 4.9.

Para um cenário possível, em um nodo da rede poderia haver uma classe de serviço EF PHB com finalidade de atender aos requisitos das aplicações que são sensíveis a características temporais, convivendo com uma classe de serviço AF PHB, a qual teria como finalidade proporcionar uma hierarquia de serviços diferenciados por seus níveis de importância. Portanto, este modelo procura transferir as propriedades de garantia de serviço obtida por uma classe no modelo GPS, para garantias de serviço entre classes em um modelo com mais de um nível, ou modelo de compartilhamento hierárquico. Sendo a sua descrição formal dada a seguir:

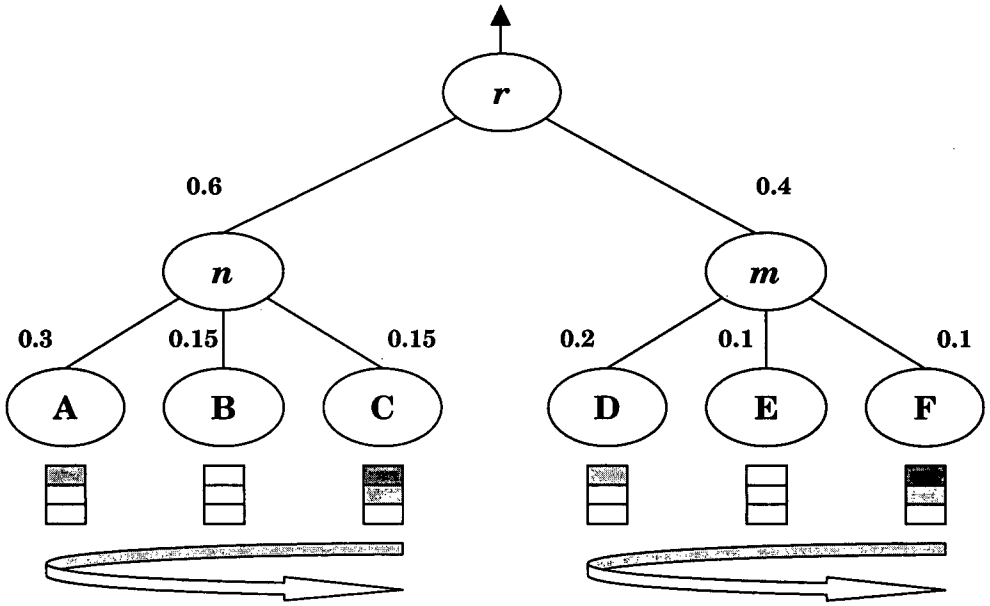

Figura 4.9 Compartilhamento fluido hierárquico entre classes de serviço.

Um escalonador pode ser representado por uma árvore a qual possui números reais positivos ϕ_n associados aos seus nodos, os quais representam a cota de compartilhamento do canal de comunicação entre classes de serviço. O nodo raiz, indicado por r, corresponde ao canal físico e cada nodo folha corresponde a um dos fluxos possíveis de uma determinada classe de serviço. Um nodo ancestral está no estado não-ocioso se no mínimo houver um nodo folha descendente que também esteja neste estado, para o intervalo de tempo $\tau = t_2 - t_1$, ou seja, no estado conservativo $B(\tau)$. Seja $W_i(t_1,t_2)$ o fluxo i de uma classe de serviço, servida no intervalo $(t_1,t_2]$, então os fluxos que compõem um agregado para esta classe pode ser dado pela equação,

$$W_n = \sum_{i \in leaf(n)} W_i(t_1,t_2) \quad para \quad i \in B(\tau) \tag{4.6}$$

onde, $leaf(n)$ é o conjunto de todos os nodos folha descendentes do nodo n. Desta forma, um escalonador hierárquico em regime conservativo é definido de forma que,

$$\frac{W_n(t_1,t_2)}{\sum_{m \in B(\tau)} W_m(t_1,t_2)} \geq \frac{\phi_n}{\sum_{m \in B(\tau)} \phi_m} \quad para \quad m \in B(\tau) \tag{4.7}$$

seja satisfeita se (a) os nodos n e m não estiverem ociosos no intervalo $(t_1,t_2]$, e (b) os nodos m e n são nodos co-irmãos os quais compartilham um mesmo nodo ancestral. Então pode-se deduzir que a equação,

$$\frac{W_n(t_1,t_2)}{\phi_n} = \frac{\sum_{m \in B(\tau)} W_m(t_1,t_2)}{\sum_{m \in B(\tau)} \phi_m} \quad para\ m \in B(\tau) \tag{4.8}$$

é satisfeita para quaisquer dois nodos co-irmãos m e n os quais não estejam ociosos no intervalo $[t_1,t_2]$.

Comparando-se (4.3) e (4.4) com (4.7) e (4.8), pode-se perceber que estas equações são muito similares. Estas equações originalmente são aplicáveis, para qualquer número de fluxos existente em um único nível no modelo fluido GPS. Da mesma forma, para o modelo hierárquico de compartilhamento, HGPS, estas são aplicáveis somente para nodos interiores de mesmo nível. Vale lembrar que fluxos (filas), neste modelo, estão associados somente com nodos folhas.

No modelo HGPS a largura total de banda r não é distribuída para todos os nodos folhas, como acontece para o modelo GPS. Ao invés disto, cada nodo interior recebe uma parcela da largura de banda de seu nodo ancestral o qual a distribui entre os seus descendentes em proporção ao compartilhamento de banda relativo entre eles mesmos. Desta forma, se dois fluxos possuírem um mesmo valor de ϕ e possuírem ancestrais diferentes, então é possível que estes recebam cotas diferentes de serviço devido a diferença das proporções relativas de seus nodos ancestrais.

Como para o modelo fluido GPS, assume-se que a soma de todas as cotas ϕ_i de todos os nodos descendentes em primeiro grau de r, seja,

$$\sum_{i \in leaf(r)} \phi_i = 1 \tag{4.9}$$

Da mesma forma para uma sub-árvore, assume-se que,

$$\sum_{i \in child(n)} \phi_i = \phi_n . \tag{4.10}$$

Destas observações segue que, $\phi_r = 1$. Dado que $r_n = \phi_n r$ também é possível demonstrar que a condição (4.5) também é aplicável para o modelo HGPS. Portanto, o modelo HGPS pode proporcionar a mesma distribuição de banda mínima e garantias de atraso para cada fluxo tal como ocorre para o modelo GPS, sendo mais bem descrito em [Bennet e Zhang 1997].

4.2.4 Escalonamento de Pacotes de um Nível (PGPS)

Um bom exemplo de uma aproximação ao modelo GPS corresponde àquele que escalona pacotes em ordem crescente de seus tempos de término previstos pelo sistema fluído, como descrito em [Demers et al. 1990] e em [Parekh e Gallager 1993].

Quando um sistema orientado por pacotes estiver pronto a escolher o próximo pacote a ser transmitido, é possível que este pacote, previsto pelo sistema fluido de referência, ainda não tenha chegado no modelo correspondente por pacotes. Esperar por este pacote requer conhecimento sobre o futuro e sempre coloca o sistema em um estado ocioso, ou não-conservativo. De forma a obter-se um sistema em regime conservativo, o escalonador de pacotes terá que escolher um pacote a ser transmitido, baseando-se somente no estado do sistema fluído de referência no intervalo de tempo τ.

No modelo *weighted fair queueing*, ou WFQ, apresentado em [Demers et al. 1990], quando o escalonador estiver pronto a transmitir o próximo pacote no intervalo de tempo τ, este seleciona um pacote, entre todos que estejam no sistema durante este tempo, que apresente o menor tempo de serviço correspondente ao tempo de serviço calculado no modelo fluido de referência, se nenhum pacote adicional estiver para chegar após o intervalo de tempo τ.

Uma implementação prática do modelo WFQ pode ser projetada com base na seguinte importante propriedade:

Propriedade 1 *A ordem relativa de término de atendimento de todos os pacotes que estejam no sistema no tempo τ é independente do tempo de chegada de qualquer pacote no sistema após o tempo τ. Ou seja, para quaisquer dois pacotes p e p' no tempo τ em um sistema fluido, se p terminar seu ciclo de serviço antes de p', assumindo-se que não haverá chegadas após o tempo τ, p terminará seu ciclo de serviço antes de p' para qualquer padrão de chegada após o tempo τ.*

Em função desta propriedade, é possível que seja mantida a ordem relativa de tempos de término de atendimento, previsto no modelo fluido de referência, para o sistema WFQ através da utilização de um mecanismo de priorização de filas. Uma implementação baseada na noção de uma função de tempo virtual é proposta em [Demers et al. 1990] e [Parekh e Gallager 1993]. Desta forma, durante qualquer intervalo de tempo $(t_1, t_2]$ não-ocioso do sistema, pode-se definir uma função de tempo virtual, com base no sistema de referência GPS, $V_{GPS}(t)$ como segue,

$$V_{GPS}(t_1) = 0 \tag{4.12}$$

$$\frac{\partial V_{GPS}(\tau)}{\partial \tau} = \frac{1}{\sum_{i \in B_{GPS}(\tau)} \phi_i} \quad \forall t_1 \leq \tau \leq t_2 \tag{4.13}$$

onde $B_{GPS}(\tau)$ é o conjunto de todas as filas não-ociosas no tempo τ. Sendo ϕ_i suas cotas de compartilhamento respectivas.

No modelo fluido, GPS, se uma conexão i apresentar demanda no tempo τ, então esta recebe uma taxa de serviço proporcional a,

$$\frac{\partial V_{GPS}(\tau)}{\partial \tau} \phi_i \tag{4.14}$$

Portanto, pode se interpretar a função de tempo virtual como o incremento da taxa marginal para a qual as filas não-ociosas do sistema recebem sua cota de serviço.

Para o *k-ésimo* pacote de um fluxo de tráfego i, seus tempos de *inicio*, S_i^k, e de *termino*, F_i^k, de um ciclo de serviço podem ser definidos como segue,

$$S_i^k = \max\left\{F_i^{k-1}, V\left(a_i^k\right)\right\} \tag{4.15}$$

$$F_i^k = S_i^k + \frac{L_i^k}{r_i} \tag{4.16}$$

onde $F_i^0 = 0$, sendo a_i^k e L_i^k o tempo de chegada de um pacote adicional e o tamanho do pacote respectivamente. Baseando-se na Propriedade 1, para todos os pacotes que se encontram no sistema WFQ no tempo τ, seus tempos relativos de ordem de *término* no sistema fluido GPS são os mesmos tempos relativos de ordem de *término* calculados pela função de tempo virtual.

Portanto o modelo WFQ pode ser implementado fazendo com que o escalonador selecione o próximo pacote a ser servido no período de tempo τ, o qual apresente o menor tempo virtual de *término* de seu ciclo de serviço.

Esta implementação tem duas vantagens: (a) o tempo virtual de término de um pacote pode ser calculado no tempo de chegada do pacote, desta forma não existe necessidade de calcular o tempo de término, no sistema GPS, toda vez que o escalonador selecionar um pacote para transmitir; (b) um mecanismo de fila priorizada, baseado no tempo de virtual de término de cada pacote pode ser utilizado para manter a ordem relativa dos tempos de término no sistema GPS, como descrito em [Bennet e Zang 1997].

4.2.5 Escalonamento de pacotes Hierárquico (HPGPS)

Aproximações de um modelo de escalonamento hierárquico por pacotes ao modelo HGPS, tal como foi obtido com relação ao modelo GPS, foram apresentadas em [Bennet e Zang 1997]. Onde este descreve que o modelo de escalonamento fluido hierárquico pode ser visto como uma integração hierárquica de múltiplos servidores GPS de um único nível. Em um escalonador GPS de um nível, existem múltiplas filas e cotas de compartilhamento de serviço predefinidas para cada fila. Durante qualquer intervalo de tempo, para qual existam filas não-ociosas, o escalador serve a estas filas em proporção de suas cotas de compartilhamento. Para o modelo HGPS, uma fila em um nodo interior corresponde a uma fila lógica, sendo que a parcela de serviço que esta receba é instantaneamente distribuída para os seus descendentes, na proporção exata de suas cotas de compartilhamento. Este processo de distribuição segue adiante na hierarquia até que sejam alcançados os nodos folhas, os quais fisicamente possuem filas associadas.

A abordagem feita por Bennet tem como objetivo a proposição de um escalonador hierárquico de pacotes denominado WF^2Q+, o qual tem por base o escalonador WFQ transposto para a sua forma hierárquica. Sendo que a sua descrição formal pode ser observada em [Bennet e Zang 1997].

4.3 Resumo

Neste capitulo procurou-se evidenciar a existência de duas categorias de mecanismos:

Mecanismos de Controle de Trafego: Os quais são responsáveis pelo condicionamento do tráfego na fronteira da rede, com base nas ações de *classificação*, *medição*, *marcação*, *conformação* e *descarte*, de maneira que este se mantenha dentro do perfil esperado para a classe de serviço.

Mecanismos de Diferenciação de Serviços: Os quais são responsáveis em implementar o comportamento esperado de uma classe abstrata de serviço de rede (PHB) em função de suas características intrínsecas definidas em termos de *vazão*, *atraso*, *variação de atraso*, *perda* e *prioridade relativa*.

Dado ao fato de que diferenciar um serviço de rede de outro em última instância signifique dividir a capacidade do canal em canais distintos, foi dado ênfase a mecanismos de diferenciação de serviços. Onde procurou-se evidenciar o modelo fluido GPS, o qual constitui um sistema de referência para algoritmos de escalonamento de um único nível hierárquico e o modelo

fluido HGPS, o qual constitui uma extensão do modelo GPS para diversos níveis hierárquicos.

Dentre todas as realizações possíveis destes modelos base, este trabalho irá utilizar o algoritmo CBQ de compartilhamento hierárquico de recursos entre classes de serviço, o qual será mais bem detalhado no capítulo 5.

Capítulo 5

Compartilhamento Hierárquico

Nodos da rede que sejam compatíveis ao modelo de Serviços Diferenciados deverão possuir uma estrutura de discriminação de serviços, associada a suas interfaces, de forma que sobre ela possam ser realizados os níveis de abstração determinados pelas Classes de Serviços ao Cliente e o seu mapeamento correspondente na forma de PHBs.

Desta forma, em um destes nodos, poderá haver somente um único PHB, uma classe de PHBs, ou uma combinação destes. Cada qual com suas características e exigências específicas a serem impostas sobre um único canal físico de comunicação. Esta configuração leva naturalmente a uma forma hierárquica de compartilhamento deste canal entre classes de serviços, passiveis de serem atribuídas a um nodo da rede.

Compartilhar, mais precisamente, significa dividir um canal físico em diversos canais lógicos. Cada um destes canais lógicos deverá representar e exibir um comportamento de acordo com um PHB específico associado.

Ainda, de forma que este processo não caracterize uma reserva de recursos pura e simplesmente a cada PHB, deverão ser identificadas quais dentre estas classes de serviços estão ativas no tempo. Sendo que o excedente de recursos provocados por classes de serviços ociosas deverá ser distribuído entre as classes de serviço ativas. Da mesma forma que em tempos de congestionamento, deverão ser assegurados as classes de serviços ativas no

mínimo as suas cotas de compartilhamento, de maneira a garantir o seu comportamento específico.

Este capítulo investiga diretrizes para a resolução deste problema, desenvolvidas por [Floyd, 1995], as quais mais tarde passaram a ser denominadas de CBQ, ou *Class-Based Queueing*. A escolha por este algoritmo neste trabalho recai por este constituir os fundamentos sobre os quais diversas variações algorítmicas são propostas e por sua aplicabilidade.

5.1 Fundamentos do método CBQ

A estrutura para compartilhamento de banda determina políticas para divisão da largura de banda de um canal em particular em tempo de congestionamento. Por exemplo, na estrutura apresentada pela Figura 5.1, o canal é compartilhado por um número de classes de tráfego, sendo que algumas das quais possuem características de tempo real enquanto outras não.

As classes de **áudio** e **vídeo** são exemplos de *classes folhas* desta estrutura e a classe **Link** constitui uma classe *interior*. A classe de **vídeo** é um exemplo de uma classe sensível ao atraso, enquanto que a classe **email** não.

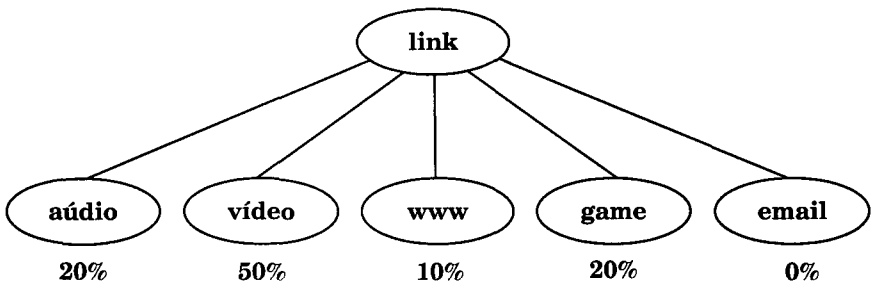

Figura 5.1 Distribuição hierárquica de recursos, em um único nível, em função de classes de serviços de rede.

Para o caso de uma estrutura planar como o desta figura, os requisitos para determinação do compartilhamento de banda são evidentes. Uma determinada parcela da largura de banda é atribuída a cada classe em termos percentuais da totalidade do canal. Estas parcelas podem ser atribuídas de forma estática, pelo operador, ou dinamicamente em função da resposta de algum algoritmo de avaliação dos recursos disponíveis.

O objetivo primário do compartilhamento de banda é que cada classe que apresente uma demanda por recurso obtenha praticamente toda a sua parcela, em algum intervalo de tempo, em momentos de congestionamento. Como conseqüência, deste procedimento, em tempo de congestionamento

algumas classes serão restringidas exatamente a sua parcela. Para classes cujo percentual seja zero, como a classe **email** no exemplo, a largura de banda recebida por esta classe em tempo de congestionamento é determinada por algum outro algoritmo na saída da rede. O mecanismo de compartilhamento não garante a esta classe qualquer parcela de banda em tempos de congestionamento.

Os objetivos do compartilhamento de banda constituem basicamente um acordo do uso da rede em termos quantitativos. Associado a estes objetivos está a noção de intervalo de tempo, sobre o qual as regras de compartilhamento devam ser aplicadas. Este intervalo poderá ser determinado através de uma constante de tempo, utilizada para estimar o tempo decorrido que cada uma das classes tenha utilizado sua parcela de banda.

No exemplo anterior, seria considerado inaceitável que as classes de áudio e de vídeo fossem impedidas de utilizar suas parcelas de banda por minutos, a cada ciclo de serviço de um escalonador. Por outro lado, para intervalos pequenos de tempo, não é necessário induzir ao escalonador um ajuste muito estreito para garantir a estas classes suas parcelas de banda.

Escalonamento baseado em níveis de prioridade poderá ser utilizado de forma a reduzir a sensibilidade ao atraso por tráfegos de tempo real, enquanto que o mecanismo de compartilhamento deve prevenir a negação total de recursos a classes de tráfego do tipo vídeo por longos períodos de tempo.

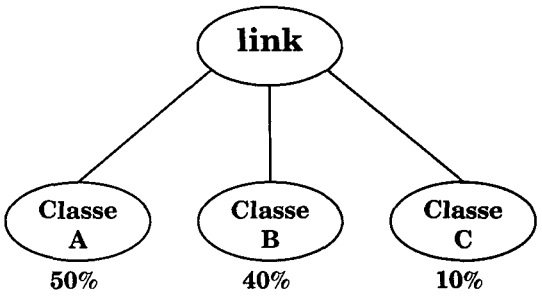

Figura 5.2 Distribuição hierárquica de recursos, em um único nível, segundo o conceito de múltiplas agências ou famílias de protocolos.

Um segundo objetivo do mecanismo de compartilhamento compreende a distribuição do excesso de recurso disponível entre as classes, sendo que esta distribuição não pode ser arbitrária e sim deverá seguir algum conjunto de regras. Para o caso de uma estrutura planar, a distribuição do excesso de recurso pode ser feita por algum mecanismo localizado no nodo de borda e isto não está especificado na estrutura de compartilhamento. Por exemplo, considerando-se o compartilhamento de banda entre duas classes de serviço, como mostra a Figura 5.2. Se a classe A emitir pouco tráfego, a classe B

poderia considerar injusto ou arbitrário, que todo o excesso de recurso disponível fosse concedido à classe C.

Para o compartilhamento de banda entre classes de serviços, o mecanismo de escalonamento poderá distribuir o excesso de recurso de forma a considerar a alocação relativa prevista na estrutura de compartilhamento para estas entidades. Múltiplas restrições ao compartilhamento de banda podem ser expressas por uma estrutura de compartilhamento hierárquica, com mais de um nível, tal como mostra a Figura 5.3.

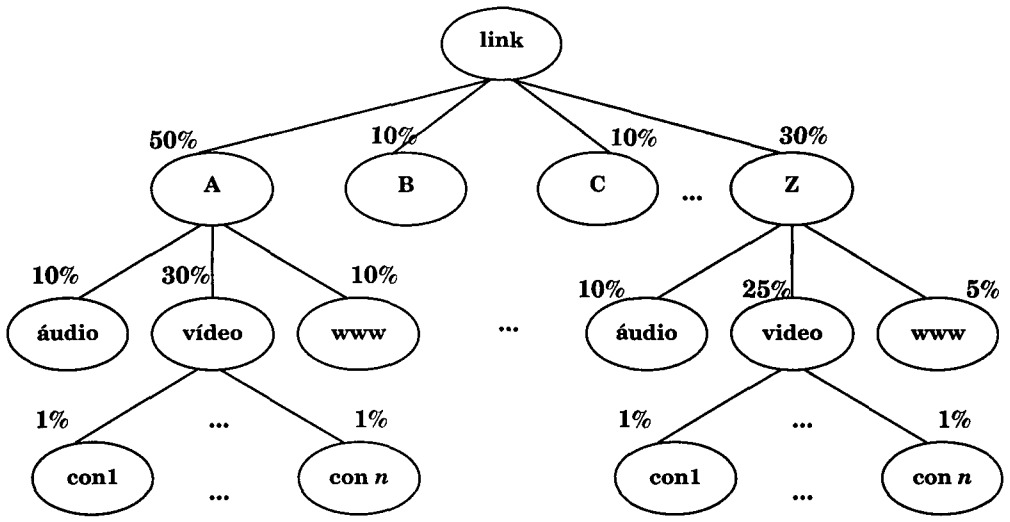

Figura 5.3 Distribuição hierárquica de recursos, em diversos níveis, entre agências ou famílias de protocolos. As conexões ocorrem somente sobre os nodos folhas.

Neste caso, filas estão associadas somente com as classes dos nodos folhas, sendo que as classes interiores são utilizadas para designar regras de utilização do excedente de recursos disponíveis. Desta forma, a aplicação de uma estrutura hierárquica de partilha assume que as classes de serviços descendentes do nodo A receberão coletivamente 50% da largura de banda efetiva do canal de comunicação, em intervalos de tempo apropriados, dado que exista demanda suficiente. Se a classe de serviço de vídeo descendente do nodo A mostrar-se pouco ativa, a estrutura hierárquica especifica que o excedente de recurso seja distribuído entre as outras classes deste mesmo nodo.

Desta forma, os objetivos do compartilhamento de banda podem ser resumidos como segue:

1: *Cada classe interior ou classe folha deverá receber basicamente sua parcela de banda alocada, sobre intervalos de tempo apropriados, dado que a demanda seja suficiente.*

2: *Se todas as classes folhas e interiores que apresentarem demanda suficiente, tiverem recebido no mínimo a sua cota de alocação prevista pelo compartilhamento de banda, a distribuição do excesso de recursos disponíveis não deverá ser arbitrário, mas deverá seguir algumas regras de imparcialidade.*

O primeiro objetivo está intrinsecamente delineado pelas restrições de compartilhamento impostas na própria estrutura hierárquica. O segundo objetivo pode ser alcançado através da aplicação de escalonadores eficientes, os quais orientados por políticas de distribuição de recursos deverão proporcionar uma distribuição do excedente de maneira imparcial entre os nodos da estrutura.

Desta forma estes objetivos requerem uma estrutura de dados associada em cada canal, descrever a estrutura de classes de cada canal, e atribuir as parcelas de banda para cada classe. Além da possibilidade de haver uma alocação dinâmica de banda entre classes existentes, a estrutura de compartilhamento de banda comporta em si mesma ambos os componentes: estático, de definição de cotas e dinâmico quanto a distribuição de excedentes.

Uma estrutura de compartilhamento de banda estática, cujas classes e parcelas de utilização de banda forem fixas, pode ser apropriada para um canal que seja compartilhado entre múltiplas classes de serviços. Neste caso, a parcela de recurso destinada a cada classe é determinada em função dos requisitos de cada classe de serviço presente em uma hierarquia e geralmente fixada por um administrador de rede.

Por outro lado, uma estrutura de compartilhamento de banda que utilize componentes dinâmicos terá que manter provisões de recursos para criação e remoção de subclasses e para proporcionar o ajuste de alocação de banda.

Uma estrutura que admitisse a criação dinâmica de classes de serviço teria que permitir a monitoração da taxa de ocupação do canal exibida em cada fluxo em tempo real. Desta forma, ter-se-ia um controle sobre cada fluxo assegurando que nenhum deles monopolize a utilização do canal. Também seria necessário algum mecanismo que limitasse o tempo de vida destas classes criadas dinamicamente [Floyd e Jacobson 1995].

5.1.1 Definições gerais

Um sistema de compartilhamento de banda assume a existência de mecanismos de escalonamento. Um *escalonador genérico* escala pacotes a partir de classes folha sem considerar as regras de alocação de banda.

Enquanto que, um *escalonador de compartilhamento* escala pacotes a partir de algumas classes folha que estejam excedendo sua cota de alocação em tempo de congestionamento. Um dos serviços do escalonador genérico é o de servir aos tráfegos de tempo real que apresentem características de atraso e de vazão particulares. Este escalonador poderá ser qualquer um entre numerosos algoritmos propostos.

Em uma hierarquia de classes, denomina-se *classe regulada*[10], a classe cujos pacotes forem escalonados na saída por um escalonador de compartilhamento. Enquanto que uma classe denomina-se, *classe não-regulada*[11], se os pacotes desta classe forem selecionados por um escalonador genérico.

Como o nome implica, um *classificador* é utilizado para classificar pacotes que se apresentam para a classe apropriada, associada ao canal de saída. Um *estimador* faz a estimação da largura de banda utilizada em cada classe, sobre um intervalo de tempo apropriado, de maneira a determinar quando, ou não, a classe tenha obtido a sua parcela de banda. A constante de tempo utilizada pelo estimador é um parâmetro crítico. Esta constante permite determinar o intervalo sobre o qual a saída tentará submeter as regras de compartilhamento sobre a estrutura de classes.

Uma classe pode entrar para o estado *sobre-limite* se recentemente estiver utilizando uma parcela de banda maior do que aquela prevista pela estrutura de compartilhamento. Entrará para o estado *sub-limite* se esta estiver utilizando uma fração da sua parcela de compartilhamento. Em último caso, uma classe estará no seu estado *limite*.

Os estados de cada classe serão determinados pelo estimador. A informação de estado de cada classe será utilizada para determinar a ação explícita que deverá ser tomada para corrigir o comportamento de partilha de recursos entre cada classe. Deve ser observado que o nodo raiz da estrutura de compartilhamento, o qual representa o próprio canal, tem para si 100% dos recursos, e, portanto, nunca poderá assumir o estado sobre-limite.

Uma classe folha poderá assumir o estado, *insatisfeita*, com o comportamento de partilha, se esta estiver no estado sub-limite e apresentar demanda reprimida por recursos[12], sendo que para os outros casos uma classe estará no estado *satisfeita*. Uma classe interior pode entrar para o estado insatisfeita se

---

[10] Aqui o termo *classe regulada* significa que esta classe necessita de uma regulagem da sua cota atual de utilização de banda. Sendo que esta regulagem deverá ser feita pelo escalador de compartilhamento.

[11] Uma *classe não-regulada* significa a classe que não está transgredindo as regras de compartilhamento, para um dado intervalo de tempo. Portanto não necessita ser regulada e será escalada pelo escalador genérico.

[12] Aqui será utilizado o termo *demanda suficiente* para evidenciar o fato de uma classe ter pelo menos um pacote a ser enviado em uma de suas filas, ou seja, *backlog persistente*.

estiver no estado sub-limite e uma de suas descendentes apresentar demanda reprimida.

Quando uma classe não estiver no estado sobre-limite ou quando não existirem classes insatisfeitas, então esta classe não necessita ser regulada pelo escalonador de compartilhamento. De outra forma, quando uma classe estiver no estado sobre-limite e houver classes insatisfeitas, então a classe que se encontra no estado sobre-limite está contribuindo para o congestionamento do canal, e deverá ser regulada, pelo escalonador de compartilhamento. Este processo de regulagem poderá prosseguir até que a classe deixe o estado sobre-limite ou até que não existam mais classes insatisfeitas.

Dadas estas diretrizes, uma classe somente será regulada quando outras classes apresentarem uma demanda reprimida e não estejam recebendo sua parcela da partilha do canal sob um determinado intervalo de tempo. Quando todas as classes estiverem satisfeitas com o comportamento de partilha, nenhuma classe necessitará ser regulada.

Todas as classes folhas em uma estrutura de compartilhamento são definidas como pertencendo ao *nível* 1, e cada classe interior possui um nível maior do que o maior nível de uma de suas descendentes.

5.1.2 Método de compartilhamento formal

Uma classe *pode continuar* no estado *não-regulada* se uma das condições for satisfeita:

1: *A classe não se encontra no estado sobre-limite, OU*

2: *A classe não possui um ancestral ao nível i que esteja no estado sobre-limite e não existem classes insatisfeitas em toda a estrutura de compartilhamento nos níveis inferiores ao nível i,*

caso contrário, a classe irá ser regulada pelo escalonador de compartilhamento.

Colocando-se estas regras na forma de uma expressão lógica então poderia-se concluir que uma classe a ser regulada deveria obedecer ao resultado lógico da expressão:

$$R = a \vee (b \wedge c) \tag{5.1}$$

onde o termo a representa a primeira diretriz e $(b \wedge c)$ a segunda diretriz.

Deve ser observado que estas diretrizes de compartilhamento são utilizadas apenas para decidir se uma classe esteja apta a ser selecionada pelo escalonador genérico, quando apresentar o estado *não-regulada*, ou quando sua parcela atual de compartilhamento de banda tenha que ser *regulada* pelo escalonador de compartilhamento.

A divisão dos recursos de banda disponíveis entre as classes não-reguladas é feita pelo escalonador genérico. Estas diretrizes são utilizadas simplesmente para determinar quando uma classe está utilizando além da sua cota de alocação de banda e, desta forma, contribuindo para o estado de insatisfação das outras classes da estrutura de compartilhamento.

Os casos a seguir, como apresentados em [Floyd e Jacobson 1995], ilustram a aplicação das diretrizes formais de compartilhamento. Para cada estrutura de compartilhamento dadas nas figuras a seguir, os círculos em negrito marcam as classes que possam estar nos estados sobre-limite e sub-limite, e pequenas filas mostram quais classes possuem uma demanda reprimida. Dado estes cenários será aplicado as diretrizes formais de compartilhamento de maneira a determinar quais classes que devam ser reguladas. Ainda, classes rotuladas com o dígito "1" são classes de tempo real e com o dígito "2" são outras classes que não de tempo real.

Verifica-se, no entanto, que a aplicação direta de (5.1) para os exemplos demonstrados por [Floyd e Jacobson 1995] não é satisfeita, a não ser na sua forma complementar, ou seja:

$$R = a \wedge (b \vee c) \tag{5.2}$$

esta lógica complementar deve referir-se a ao fato de que uma classe *não* será regulada, enquanto que as diretrizes formais determinam os casos para os quais *serão* reguladas (lógica positiva).

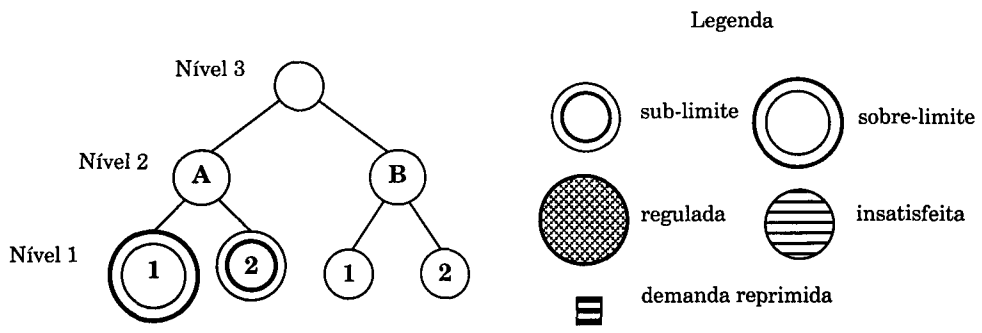

Figura 5.4 Caso 1, verificação de estado e determinação de quando uma classe deva ser regulada segundo as regras formais de compartilhamento.

Para o caso 1, ilustrado pela Figura 5.4, observa-se que nenhuma classe está insatisfeita, então segundo as regras de compartilhamento formais, nenhuma classe deverá ser regulada.

Classe	*a*	*b*	*c*	$R = a \wedge (b \vee c)$
A(1)	1	0	1	1
B(1)	1	0	1	1

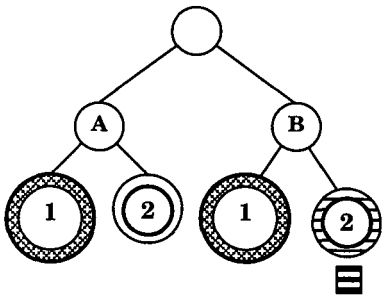

Figura 5.5 Caso 2, segundo as regras de compartilhamento formais, as classes de tempo real das agências A e B deverão ser reguladas.

Para o caso 2, ilustrado pela Figura 5.5 existem duas classes que estão no estado sobre-limite, mas somente a classe 2 da agência B esta insatisfeita e a classe da agência A não está no estado sobre-limite. A partir das diretrizes de compartilhamento, ambas as classes de tempo real poderiam permanecer não-reguladas não fosse o estado de insatisfeita, com demanda reprimida, apresentado pela classe 2 da agência B, portanto, neste caso ambas as classes de tempo real serão reguladas.

Classe	*a*	*b*	*c*	$R = a \wedge (b \vee c)$
A(1)	1	1	1	1
B	0	0	1	0

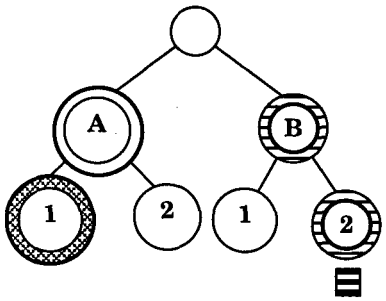

Figura 5.6 Caso 3, segundo as regras de compartilhamento formais, somente a classe de tempo real da agência A deverá ser regulada.

Para o caso 3, ilustrado pela Figura 5.6, a classe da agência A e a classe de tempo real da agência A estão no estado sobre-limite, enquanto que a classe da agência B e a classe 2 da agência B estão insatisfeitas. A partir das regras de compartilhamento, a classe de tempo real da agência A necessita ser regulada. Devido a que esta classe transgride ambas as regras do compartilhamento formal, pois está no estado sobre-limite, possui um ancestral neste estado e existe na estrutura classes insatisfeitas nos níveis inferiores ao nível *i*.

Classe	*a*	*b*	*c*	$R = a \wedge (b \vee c)$
A(2)	1	0	0	0
B(1)	1	1	0	1

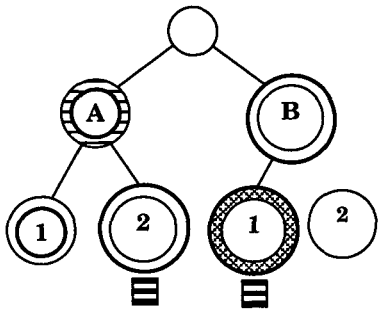

Figura 5.7 Caso4, segundo as regras de compartilhamento formais, somente a classe de tempo real da agência B deverá ser regulada.

Para o caso 4, ilustrado pela Figura 5.7, nenhumas das classes folhas estão insatisfeitas, mas a classe da agência A está insatisfeita. Devido a que a classe da agência A está no estado sub-limite e não existe nenhuma classe abaixo dela insatisfeita, esta permanecerá desregulada. No entanto, de acordo com as diretrizes de compartilhamento a classe de tempo real da agência B deverá ser regulada, pois encontra-se no estado sobre-limite e possui um ancestral no mesmo estado.

Deve ser observado que não foi especificado a freqüência para a qual o escalonador deva verificar o estado das classes para então decidir sobre suas regulagens. Uma alternativa seria fazer esta verificação imediatamente antes de enviar um pacote a partir de uma classe. Mas ainda pode ser sugerido que esta verificação seja feita com menor freqüência.

Quando o controle sobre uma classe deva ser considerado como terminado? Uma das possibilidades é a de que uma classe regulada deveria permanecer sob a ação de regulagem enquanto as diretrizes formais de compartilhamento não estiverem satisfeitas.

Uma classe pode oscilar freqüentemente entre os estados *regulada* e *não-regulada*, mas isto necessariamente não constitui um problema. Se por questões de implementação for considerado necessário reduzir a freqüência destas oscilações, as seguintes diretrizes alternativas poderiam ser utilizadas:

5.1.2.1 Métodos formais alternativos

Uma classe pode continuar no estado, *não-regulada*, se uma das seguintes condições for encontrada:

1: *A classe não encontra-se no estado sobre-limite, OU*

2: *A classe não possui um ancestral no nível i que esteja no estado sobre-limite, e toda a estrutura de compartilhamento não possui classes insatisfeitas nos níveis inferiores ao nível i,*

caso contrario, a classe será regulada pelo escalonador de compartilhamento.

Uma classe que deva ser regulada irá continuar a ser *regulada* até que uma das seguintes condições for satisfeita:

1: *A classe está no estado sub-limite, OU*

2: *A classe possui um ancestral no nível i que esteja no estado sub-limite, e toda a estrutura de compartilhamento não possui classes insatisfeitas em níveis inferiores ao nível i.*

Uma estrutura de compartilhamento poderá marcar algumas das classes para serem *classes isentas* ou *limitadas,* se for necessário. A noção de tráfego isento implica que o tráfego nunca será restringido pelo escalonador de sua parcela de recursos alocados, não importando o nível de congestionamento do canal de saída[13].

Uma classe quando isenta poderá assumir que possui uma cota de compartilhamento de 100% da largura de banda disponível. Para esta classe o escalonador genérico e o mecanismo de controle de admissão deverão garantir que o tráfego proveniente desta classe não viole os objetivos de compartilhamento, ou deverá haver um entendimento claro que escalonar este tipo de tráfego tem precedência sobre os objetivos de compartilhamento.

A uma *classe limitada* não é permitido que empreste recursos de suas classes ancestrais, não importando o estado limite destas classes. Isto pode ser feito, por exemplo, para uma classe de tráfego que consiste de uma única conexão de tempo real de alta prioridade, onde um valor baixo na variação do atraso é mais importante do que um valor baixo do atraso médio.

Em uma implementação da estrutura de compartilhamento, cada classe poderá conter um campo denominado de *ancestral*, concedendo a classe seu grau de parentesco, e um campo denominado de *empréstimo*, indicando quando esta classe poderá ou não emprestar recursos de suas classes parentes. Uma classe do tipo *limitada* terá seu campo *empréstimo* ajustado para não permitir empréstimos.

Uma classe *isolada* não permite que classes que não sejam suas descendentes façam empréstimos de seus próprios recursos, e a mesma, por sua vez, não empresta recursos de outras classes. Uma classe isolada irá manter o campo

[13] Propõe se que tráfegos de tempo real sejam marcados como isentos, e procedimentos de controle de admissão sejam os únicos mecanismos a assegurar que o tráfego não viole as regras de compartilhamento.

ancestral vazio, e simplesmente será associada a ela uma fração da largura de banda.

5.1.3 Métodos de compartilhamento aproximados

A seção anterior descreveu um conjunto de diretrizes para o compartilhamento formal. Portanto, a partir destas diretrizes formais, a decisão de quando ou não regular uma classe, depende não somente do estado limite de suas classes ancestrais, mas também do estado de satisfação das outras classes presentes na estrutura de compartilhamento.

É possível que estas diretrizes possam ser implementadas com eficiência dada a escolha de arquiteturas apropriadas. Mesmo assim, será necessário considerar outras aproximações as diretrizes de compartilhamento, as quais proporcionem uma implementação mais imediata e eficiente.

Esta seção discute dois métodos aproximados ao modelo de compartilhamento de recursos formal: a primeira aproximação denomina-se de método de compartilhamento *Ancestor-Only*, enquanto que a segunda aproximação denomina-se *Top-Level*.

No método *Ancestor-Only*, por facilidade de implementação, a decisão de quando uma classe deva ser regulada é determinada somente com base no estado limite da própria classe e de suas ancestrais. Quando este método for aplicado a uma estrutura de compartilhamento planar, como o da Figura 5.1, pode trazer algumas dificuldades.

Pois, uma classe folha que esteja no estado sobre-limite não pode permanecer no estado, não-regulada, quando a classe raiz estiver em seu limite, e a capacidade total do canal tenha sido encontrada. Se este for o caso, então uma classe que esteja no estado sobre-limite nunca poderia ser regulada. Uma resposta direta a esta questão seria a de permitir que uma classe que encontra-se no estado sobre-limite poderá permanecer não-regulada somente quando alguma classe ancestral estiver no estado sub-limite, ao invés de simplesmente estar em um estado não-sobre-limite.

5.1.3.1 Método *Ancestor-Only* de compartilhamento

Uma classe poderá permanecer no estado, *não-regulada*, se uma das condições seguintes for satisfeita:

1: *A classe não está no estado sobre-limite, OU*

2: *A classe possui um ancestral no estado sub-limite,*

caso contrário, a classe será regulada pelo escalonador de compartilhamento.

Uma deficiência deste método ocorre quando o estado de satisfação das classes co-irmãs não for examinado. Neste caso, nenhuma distinção pode ser feita entre as classes de tempo real da agência A, descrito pela Figura 5.4, com as classes de tempo real da agência B, descrito pela

Figura 5.5. Neste cenário, as classes de tempo real da agência A, da Figura 5.4, serão reguladas desnecessariamente, em função das regras deste método, ou nenhuma das classes de tempo real serão reguladas e os objetivos do compartilhamento de recursos não serão satisfeitos.

Embora o método *Ancestor-Only* possa proporcionar resultados aceitáveis para a maioria dos casos, ainda assim, não constitui um método formal robusto de compartilhamento de recurso.

Devido a que o estimador faz distinção entre classes ancestrais que estejam nos estados limite e sub-limite. Isto faz com que uma classe folha, que esteja no estado sobre-limite, permaneça não-regulada somente quando houver um ancestral no estado sub-limite. Este método é sensível ao caráter quantitativo dos parâmetros utilizados para distinguir entre classes que possam estar nos estados limite e sub-limite.

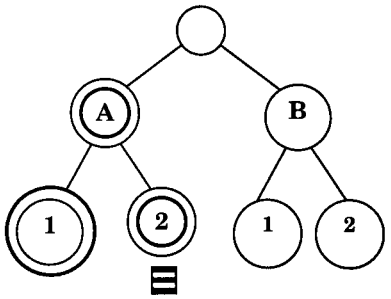

Figura 5.8 Sensibilidade do método *Ancestor-Only* a quantificação paramétrica utilizada para distinguir os estados *limite* e *sub-limite*.

Também pode ser sensível a outros parâmetros do estimador. Considerando a estrutura de compartilhamento da Figura 5.8, assumindo-se que após um período de tempo a agência B tenha utilizado toda a banda disponível e que a classe de prioridade 1 da agência A tenha recentemente transmitido uma rajada de pacotes, sendo rotulada pelo estimador para o estado sobre-limite.

Neste cenário, as classes de tempo real da agência A irão continuar como aptas a transmitir de forma não-regulada, enquanto o estimador continuar a rotular a própria classe A para o estado sub-limite, não importando o estado de insatisfação da classe de prioridade dois da agência A.

Portanto, este método é sensível ao aporte máximo, de pacotes, que pode ser transmitido antes que seja possível determinar que uma classe interior esteja no estado sobre-limite.

5.1.3.2 Método *Top-Level* de compartilhamento

As diretrizes para o método *Top-Level* apresentam pequenas modificações com relação ao método *Ancestor-Only*, e lhe conferem uma aproximação mais robusta ao método formal de compartilhamento. Pelas regras deste método, o sistema continua por examinar o estado limite das classes ancestrais. Ainda, além de cobrir as regras do método *Ancestor-Only*, considera o estado limite de algumas classes até um determinado nível de profundidade determinado por uma variável denominada de *Top-Level*.

O sistema mantém a variável *Top-Level*, a qual indica o maior nível a partir do qual permite-se que uma classe possa fazer empréstimos de recursos. Tal como no método formal de compartilhamento, onde não se permite que seja feito empréstimo a partir das classes que estejam no nível i ou acima, se houverem classes insatisfeitas no nível i-1, este método utiliza-se a variável *Top-Level* para indicar o maior nível a partir do qual uma classe possa fazer empréstimos, bem como, várias heurísticas para atribuir o valor desta variável.

Uma classe pode permanecer no estado *não-regulada* se uma das seguintes condições for satisfeita:

1: *A classe não esta no estado sobre-limite, OU*

2: *A classe possui um ancestral que esteja no estado sub-limite, cujo nível seja no máximo o valor de Top-Level.*

caso contrário, a classe será regulada pelo escalonador de compartilhamento.

Isto abre uma gama de possibilidades para determinação heurística do valor de *Top-Level*. Se para esta variável for atribuído um valor infinito, então este método comporta-se como o método *Ancestor-Only*.

Quando esta variável assumir um valor correspondente ao nível mínimo para o qual exista uma classe insatisfeita, este método comporta-se essencialmente como o método formal de compartilhamento. Por exemplo, se o sistema atribuir o valor 1 para *Top-Level*, quando houver uma classe que não esteja no estado sobre-limite e possui uma fila que não está vazia, então enquanto

Top-Level permanecer em 1, somente classes que não estiverem no estado sobre-limite serão capazes de transmitir pacotes.

De qualquer modo, ao invés de fazer uma implementação precisa do método formal de compartilhamento, o que acarretaria em uma sobrecarga de processamento, o método *Top-Level* permite uma aproximação estreita a este método e o valor heurístico de atribuição a variável *Top-Level* permite diminuir a sobrecarga de processamento.

Heurísticas de atribuição de valores a variável *Top-Level*

1: Se um pacote chegar para uma classe que esteja no estado não-sobre-limite, então atribui-se o valor 1 para *Top-Level*.

2: Se o valor de *Top-Level* for igual a i, e um pacote chegar para uma classe que esteja no estado sobre-limite, a qual possui uma classe ancestral no estado sub-limite em níveis inferiores a i, diga-se j, então atribui-se o valor j para *Top-Level*.

3: Depois que um pacote foi transmitido a partir de uma classe e esta classe agora possui uma fila vazia ou é incapaz de continuar no estado não-regulada, então é atribuído um valor infinito para *Top-Level*.

Seguindo-se estas diretrizes, o sistema irá atribuir o valor i para *Top-Level* somente quando este souber que algumas classes estão aptas a transmitir sem que seja necessário fazerem empréstimos de um de seus ancestrais acima do nível i.

O ajuste do valor de atribuição a variável *Top-Level* reflete a necessidade de um conhecimento parcial sobre o sistema. Por exemplo, a partir destas diretrizes a variável *Top-Level* deverá ser maior do que 1 mesmo que exista uma classe não-sobre-limite que possua uma fila não-vazia.

Mesmo que este método requeira uma sobrecarga de processamento para manter o valor da variável *Top-Level*, quando comparado ao método *Ancestor-Only*, existem casos para os quais este método requer menos processamento do que para o próprio método *Ancestor-Only*. Por exemplo, quando o valor da variável *Top-Level* for 1, o escalonador não necessita verificar o estado limite das classes ancestrais antes de decidir quando ou não uma classe deva ser regulada.

5.1.4 O estimador

O estimador determina o estado limite das classes presentes na estrutura de compartilhamento. Os dois parâmetros chave para o estimador são a *constante de tempo* e a *freqüência* na qual o valor do estado limite das classes é atualizado. A constante de tempo pode ser um parâmetro explicito de projeto junto ao sistema. Pode admitir-se que o sistema deverá recalcular o estado limite para uma classe e seus ancestrais, logo após ter sido transmitido um pacote desta classe. O estimador utiliza uma média exponencial balanceada móvel, EWMA, tal como utilizada pelo TCP para calcular o atraso RTT.

Este estimador procura por tempos de partida recentes entre-pacotes, utilizando um peso de valor decrescente, a medida para a qual os pacotes tornarem-se distantes, e indiretamente calcula a média do tempo de partida entre-pacotes, ou, de forma recíproca, a taxa média de pacotes.

Seja s o tamanho do pacote recentemente transmitido em termos de bytes, seja b a parcela de banda atribuída a uma das classes em termos de bytes por segundo e seja t o tempo medido entre o pacote recentemente transmitido e um novo pacote que chegue para a mesma classe, como mostra a Figura 5.9.

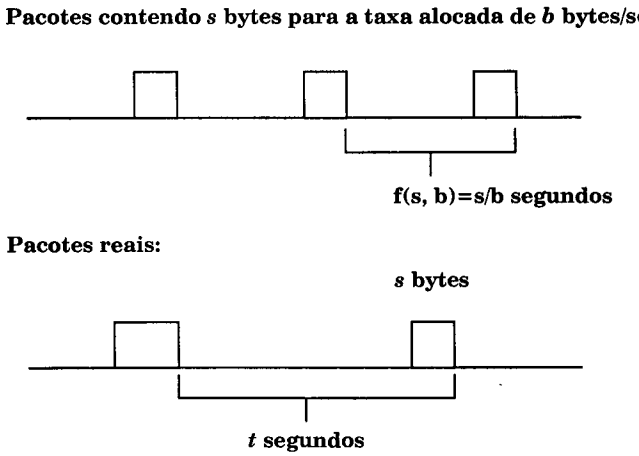

Figura 5.9 Definição de variáveis para o cálculo e conseqüente determinação dos estados limites de uma classe.

Se o sistema passar a transmitir pacotes de tamanho s a partir de uma classe, a qual possui uma largura de banda alocada precisamente no valor b, então o tempo decorrido entre dois pacotes sucessivos será,

$$f(s,b) = s/b \text{ , segundos.} \qquad (5.3)$$

e,

$$\Delta t = t - f(s,b) \tag{5.4}$$

será a discrepância entre o tempo de *entre-partida real* com o tempo de *entre-partida alocado* para esta classe. Deve-se observar que Δt é negativa quando a classe estiver excedendo a sua cota de compartilhamento, caso contrário será positiva. O cálculo da média *avg*, da variável Δt, utiliza a seguinte equação,

$$avg \leftarrow (1-w)avg + w * \Delta t. \tag{5.5}$$

A partir de valores apropriados destes parâmetros, e sendo o peso w escolhido como uma potência de dois negativa. Para o cálculo de Δt, a função *f(s,b)* pode ser calculada explicitamente, ou pode ser determinada utilizando o tamanho do pacote s como um índice para um vetor associado a classe.
O peso w determina a constante de tempo do estimador. Se a taxa de transferência para uma classe mudar repentinamente, causando também uma mudança brusca do valor calculado da variável Δt, então levará $-1/\ln(1-w)$ pacotes a serem transferidos a partir desta classe, antes que o valor calculado de *avg* mova-se para 63% da distância do antigo valor de Δt para o seu novo valor [Young 1984]. Isto corresponde a uma constante de tempo de aproximadamente,

$$\frac{-s}{b\ln(1-w)}, \text{ segundos} \tag{5.6}$$

para que uma classe transmita s-bytes de cada pacote próximo ao valor da cota de compartilhamento b alocado para a classe em termos de bytes por segundo.

O estimador não deverá permitir que uma classe previamente ociosa transmita uma rajada de tráfego muito intensa, antes que seja determinado se esta classe esteja no estado sobre-limite. De outra forma, se uma classe tenha utilizado uma fração muito pequena da sua cota de compartilhamento, então é permitido a esta classe um valor maior para o seu parâmetro *avg*.

Uma classe que se encontrava ociosa, a qual possui uma cota de alocação de b bytes por segundo e um valor A para a sua média *avg*, poderá transmitir n s-bytes de cada pacote antes que seja estimado seu estado sobre-limite[14], sendo

[14] Sendo l a largura de banda do canal em *bytes* por segundo.

$$n \leq \left\lceil \frac{\log\left(\frac{A}{s/b - s/l} + 1\right)}{-\log(1-w)} \right\rceil . \qquad (5.7)$$

Assim, limitando o máximo valor que a variável *avg* possa assumir, o estimador pode limitar o número de pacotes que uma classe ociosa possa transmitir antes que esta seja promovida para o estado sobre-limite.

A decisão de implementação do estimador deveria explicitamente considerar a extensão que o estado limite de uma classe possa ser influenciado pela cota de largura de banda que uma classe tenha recebido em excesso. Isto pode ser feito através de um parâmetro que especifica um valor mínimo negativo para a variável *avg*.

Deve ser observado que um estimador explicitamente não estima a largura de banda utilizada por uma classe. A função principal do estimador é a de indicar com precisão e imparcialidade quando uma classe está além ou aquém de seu limite de cota, sendo que o valor exato de *avg* calculado pelo estimador pode ser sensível a fatores externos tal como o tamanho de pacote de uma classe.

Uma implementação alternativa para o estimador poderia considerar que o sistema a cada t segundos irá recalcular o estado limite de cada classe tendo decorrido T segundos da última operação. Esta técnica poderá ser adequada para $t << T$. Com esta implementação, a precisão do sistema em satisfazer os objetivos de compartilhamento de recursos está limitada pela razão entre T e t. O valor de T é determinado pelo intervalo de tempo desejado, sobre o qual os objetivos de compartilhamento devam ser aplicados. Dado um valor de T, diminuindo o valor de t aumenta a precisão para a qual o sistema satisfaz os objetivos de compartilhamento de recursos.

Para o método formal de compartilhamento de recursos, e para as classes folhas dos métodos de compartilhamento *Ancestor-Only* e *Top-Level*, o escalonador necessita saber quando ou não uma classe esteja no estado sobre-limite. Neste caso o estimador utiliza o valor real da cota de alocação b da classe para calcular $f(s/b)$, ou seja, o tempo decorrido entre-partida de pacotes.

De outra forma, para classes interiores dos métodos *Ancestor-Only* e *Top-Level*, o escalonador necessita saber quando ou não uma classe esteja no estado sub-limite. Para isto, o estimador utiliza um valor de banda b' o qual é um pouco menor do que o valor de cota b alocado para a classe, para prosseguir com o cálculo de $f(s,b')$.

O valor de *avg* calculado pelo estimador é então utilizado para indicar o estado limite de uma classe, sendo este armazenado no campo *tempo-de-transmissão* associado à classe. Este campo indica a próxima vez que o sistema estará habilitado a transmitir um pacote a partir desta classe. Para qualquer classe que apresente um *avg* positivo, o estimador atribui o valor do campo *tempo-de-transmissão* para zero, indicando que a classe encontra-se no estado sub-limite.

Para uma classe não-regulada e com *avg* negativo, por exemplo uma classe interior, é atribuído ao campo *tempo-de-transmissão* um valor de x segundos adiante do tempo corrente, sendo

$$x = -avg\frac{1-w}{w} + f(s,b), \tag{5.8}$$

onde s é o tamanho do pacote que está sendo transmitido a partir desta classe. Se o sistema aguardar ao menos x segundos antes de transmitir outro pacote desta mesma classe, então esta classe não estará mais no estado sobre-limite.

Para uma classe regulada que possua um valor de *avg* negativo, por exemplo uma classe folha não-isenta, o escalonador de compartilhamento irá atribuir ao campo *tempo-de-transmissão* associado a classe o valor de *f(s,b)* segundos adiante do tempo corrente. Este valor constitui o tempo antecipado para o qual uma classe estará apta a transmitir um pacote. Desta forma, uma classe regulada nunca é restringida pelo escalonador de compartilhamento a um valor menor do que sua cota de compartilhamento, não importando o excesso de banda que esta tenha utilizado no passado.

5.1.5 O escalonador genérico

O escalonador genérico faz o escalonamento de pacotes a partir de classes não-reguladas. Em uma das implementações possíveis pode-se considerar que o sistema mantenha uma fila separada para cada classe associada ao canal de saída.

Após cada classe transmitir pacotes para o ponto de saída da rede, o escalonador genérico, neste ponto, decide qual classe poderá transmitir um próximo pacote para a saída do canal. O escalonador primeiramente faz o escalonamento de pacotes das classes de maior prioridade. Para classes de mesma prioridade, o escalonador poderá utilizar uma variante do método WRR, onde os pesos são proporcionais à cota de alocação de banda de cada classe. Estes pesos determinam o número de bytes que são permitidos a cada classe transmitir em cada ciclo de serviço do escalonador. Quando uma classe transmite um número de bytes além de sua cota a cada ciclo, devido a que

pacotes não são desmontáveis em termos de bytes, então esta classe tem a sua cota de transmissão de bytes reduzida no próximo ciclo.

A utilização da técnica WRR para o escalonador serve a dois propósitos. O primeiro é o de assegurar a classe de maior prioridade que esta receba sua cota de alocação de banda mesmo sobre intervalos mínimos de tempo. Se no máximo for alocada metade da largura de banda do canal para esta classe, então esta receberá sua cota de alocação, se apresentar demanda suficiente, a cada ciclo de serviço do escalonador.

O segundo propósito é o de assegurar que a largura de banda distribuída entre classes não-reguladas de mesma prioridade, seja feita em proporção as suas cotas de alocação de banda previstas na estrutura de compartilhamento. Como discutido anteriormente, a distribuição do excesso de banda disponível entre outras classes não deve ser arbitrária, mas sim seguir um conjunto de regras apropriadas. Portanto, a utilização de um escalonador genérico baseado em níveis de prioridade, o qual utilize o método *weighted round robin* entre estes níveis de prioridade, resulta no excesso de banda disponível sendo distribuída entre as classes de maior prioridade de maneira proporcional as suas cotas relativas de alocação de banda.

Antes que o escalonador genérico transmita um pacote a partir de uma classe, este verifica o estado limite da classe simplesmente comparando o campo *tempo-de-transmissão* associado à estrutura com o tempo atual. Se o campo *tempo-de-transmissão* for zero, então a classe poderá estar nos estados limite ou sub-limite, e o escalonador genérico estará apto a transmitir um pacote a partir desta classe. Se o valor do campo *tempo-de-transmissão* for não nulo e sim menor do que o tempo atual, então a classe deve estar no estado sobre-limite e o escalonador ainda esta apto a transmitir um pacote a partir desta classe.

Se o valor do campo *tempo-de-transmissão* para uma classe for maior do que o tempo atual, então a classe encontra-se no estado sobre-limite. Neste caso, o escalonador genérico pode somente transmitir um pacote a partir desta classe se as diretrizes de compartilhamento permitirem. Por exemplo, no método *Ancestor-Only*, se o valor de *tempo-de-transmissão* for maior do que o tempo corrente, então o escalonador genérico pode somente transmitir um pacote a partir desta classe se esta possuir uma classe ancestral no estado sub-limite.

Um requisito essencial para qualquer proposição de um escalonador genérico é que o escalonador projetado tenha como objetivo uma implementação eficiente. Em [Wakeman et al. 1995] propõe-se uma implementação eficiente de um mecanismo CBQ.

5.1.6 O escalonador de compartilhamento

O escalonador de compartilhamento controla o escalonamento de pacotes a partir de classes reguladas. Este escalonador trabalhando em conjunto com o escalonador genérico mantém as classes reguladas aos seus limites de compartilhamento de banda.

Uma implementação possível poderia utilizar dois métodos para manter as classes reguladas dentro de seus limites. O primeiro método utiliza o campo *tempo-de-transmissão* associado a uma classe como indicação da necessidade de regulagem desta classe. Um segundo método poderia considerar os casos nos quais uma classe é forçada a permanecer ociosa por um tempo substancial de transferência de pacotes. Este método envolve a remoção temporária da classe da lista encadeada de classes, no mesmo nível de prioridade, para voltar a inseri-la após a expiração de um intervalo de tempo.

Quanto ao primeiro método, o escalonador de compartilhamento atribui ao campo *tempo-de-transmissão*, associado a uma classe regulada, o valor da função $f(s, b)=s/b$ segundos adiante do tempo corrente, dado que um pacote transmitido contenha s bytes e que a classe possui uma cota de compartilhamento de b bytes por segundo. O resultado disto é que o escalonador genérico considera uma classe no estado sobre-limite até o tempo indicado no campo *tempo-de-transmissão*. Quando este tempo expirar, o escalonador genérico estará apto a transmitir um pacote a partir desta classe, não importando o valor da variável *avg* ou o estado limite de suas classes ancestrais. Se a classe se mantiver no estado sobre-limite após a transmissão de um pacote, como indicado pela variável *avg* associada à classe, então ao campo *tempo-de-transmissão* novamente é atribuído o valor $f(s, b)=s/b$ segundos adiante do tempo corrente.

Para uma classe regulada, o escalonamento exato de pacotes a partir desta classe ainda é determinado pelo escalonador genérico. Para o caso de uma classe de alta prioridade o escalonador genérico atua como se tivesse transmitindo um pacote de uma classe regulada após a expiração de tempo descrito no campo *tempo-de-transmissão*. Para uma classe de baixa prioridade o escalonador genérico poderá ser atrasado antes que venha a transmitir um pacote de uma classe regulada. Se isto ocorrer com freqüência, então a variável *avg* mudará de sinal, indicando que a classe que estava no estado sobre-limite agora não encontra-se mais neste estado e a classe não será mais regulada pelo escalonador de compartilhamento. Neste ponto, o valor atribuído ao campo *tempo-de-transmissão* será nulo.

5.2 O método HTB

O método HTB, *Hierarchical Tocken Bucket*, desenvolvido por [Devik, 2001] é um algoritmo variante do método CBQ. A principal diferença entre os métodos reside na utilização de um mecanismo de token Bucket para determinação dos estados limites de uma classe de serviço, ao invés de utilizar o estimador baseado na média exponencial móvel EWMA.

Este método utiliza uma variável, L, para identificar o nível da estrutura a partir do qual uma classe esteja apta a transmitir, tal como para o método *Ancestor-Only*. Inicialmente o valor da variável L possui o valor 1 e a diretriz única deste método, a seguinte:

Uma classe folha pode continuar no estado não-regulada somente se existir no nível L classes ancestrais a este nível que não estejam no estado sobre-limite.

Para o caso em que todas as folhas estiverem no estado sobre-limite, L é incrementado, o que significa que este nível está impedido de transmitir. O processo de incremento de L continua até que seja encontrado um nível a partir do qual seja possível transmitir um pacote, ou a raiz da hierarquia seja encontrada.

O estimador, neste método, é um mecanismo de token bucket. A determinação de estado de uma classe é feita em função do número de créditos retirado do token bucket. Sendo que a variação de sinal nos créditos, de positivo para negativo, determina o estado de cada classe.

Um segundo mecanismo de token é utilizado, denominado de *teto*. Este é utilizado com o objetivo de impor o nível limite, para o qual uma classe possa fazer empréstimos. Desta forma, uma classe poderá fazer empréstimos enquanto "tiver teto" (créditos no segundo token bucket). Ajustar o primeiro e o segundo token bucket para um mesmo valor equivale a determinar que uma classe é to do tipo limitada.
Portanto, tornando o conceito de classe limitada e não limitada mais intuitivo e permitindo seja estabelecido o nível de empréstimo que uma classe possa receber.

5.3 Resumo

Neste capítulo procurou-se evidenciar a lógica e os conceitos relacionados ao compartilhamento hierárquico de recursos entre classes de serviço. Também, procurou-se ter como documento base de referência o trabalho desenvolvido

em [Floyd e Jacobson, 1995], por este constituir uma das origens da proposição de mecanismos para diferenciação de serviços e devido a sua aplicabilidade.

Neste capitulo foram introduzidos os seguintes conceitos fundamentais: *classes interiores*, *classes folhas*, *níveis hierárquicos*, *estados de uma classe* e proposições de modelos de compartilhamento aproximados ao modelo formal, tais como: o método *Ancestor-Only,o* método *Top-Level* e o método *HTB*.

Maiores informações sobre a parametrização de um mecanismo CBQ podem ser encontradas em [Floyd, 1995].

Capítulo 6

Ambiente e Plataforma de Teste

Este capítulo tem como objetivo principal verificar a aplicabilidade dos conceitos de diferenciação de serviços sobre o modelo IP, utilizando-se o conjunto de ferramentas disponíveis no sistema operacional Linux. Baseado neste sistema será instalado um roteador nas dependências do provedor de acesso a Internet, CNX. Como exposto na RFC 2474, os conceitos bases para o modelo de Serviços Diferenciados não devem ser dependentes de aplicação e devem possibilitar a utilização dos mecanismos atualmente disponíveis. Desta forma, serão determinadas classes de serviços internas ao provedor com base na classificação de pacotes, utilizando-se propriedades multi-campo de pacotes IP, possibilidade esta também prevista pela RFC 2474.

Como diferenciar implica diretamente em dividir o canal de comunicação em canais que apresentem especialidades específicas, o maior objetivo destes experimentos, localiza-se, portanto, na investigação do comportamento geral do modelo IP quando submetido a restrições impostas por um algoritmo de compartilhamento do canal de comunicação. Base para a qual serviços genuinamente compatíveis ao modelo de Serviços Diferenciados poderão então ser construídos.

Neste caso, será verificado o comportamento do modelo IP quando submetido aos algoritmos de compartilhamento de banda CBQ e HTB. Caso seja demonstrado um comportamento viável, então estará aberta uma gama de

possibilidades e soluções para se alcançar a diferenciação de serviços sobre o modelo IP.

6.1 Controle de Trafego no Linux

O sistema operacional Linux é uma variante, o qual adota um modelo de fonte aberta, do sistema operacional UNIX, iniciado por Linuz Torvalds em 1991 e continuamente desenvolvido pela comunidade de softwares de fonte aberta, desde então.

Seu código fonte esta livremente disponível sob a forma de licença GNU. Sob este aspecto e dado ao fato de que este sistema apresenta alta flexibilidade junto aos blocos de código referentes aos mecanismos de rede, os quais permitem configurar roteadores que apresentem funções de estado-da-arte, que se formaram as razões principais da sua escolha como plataforma base para implementação do ambiente de teste.

A parte do sistema operacional Linux, a qual é responsável pelo compartilhamento de banda e escalonamento de pacotes, é denominada de *Traffic Control (TC)*, estando disponível desde a versão do kernel 2.1.90. Conjuntamente a versão do kernel 2.3 este recurso tornou-se parte integrante do sistema. Sendo esta descrição, baseada na versão do kernel 2.4.13, a qual constitui a versão mais recente no momento em que este texto foi escrito.

A maior parte do código para controle de tráfego desta versão (incluindo Serviços Integrados e RSVP) foi escrita por [Alexey Kuznetsov, xxx]. Mais tarde foram adicionados suportes para o modelo de Serviços Diferenciados por [Werner Almesberger, xxx]. Embora o conjunto de ferramentas para controle de tráfego no Linux esta centrada para a conformação do tráfego de saída, este também possui mecanismos para policiamento de tráfego.

Como para a maioria dos sistemas operacionais UNIX, a implementação do código de rede segue um modelo simplificado daquele previsto para o modelo OSI, como mostra a Figura 6.1. No princípio, os módulos de código referentes a rede, originalmente foram derivados do código fonte do sistema BSD Unix, sendo completamente reescritos para a versão do kernel 2.2, em busca de melhor desempenho.

Nas camadas apresentadas na pilha de protocolos IP, somente a camada de enlace, de rede e de transporte são executadas no espaço de endereçamento do sistema, enquanto que a camada da aplicação executa no espaço de endereçamento do usuário.

O escalonamento de pacotes e controle de tráfego é implementado entre as camadas de enlace e de rede. Desta forma, mostram-se independentes de protocolo e do dispositivo de rede.

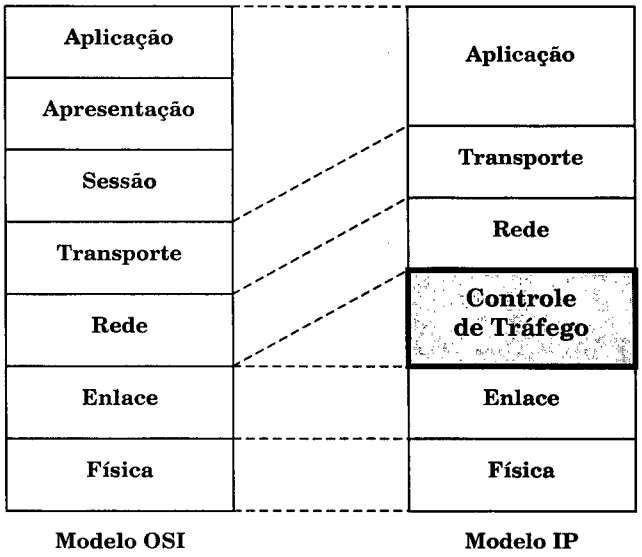

Figura 6.1 Posicionamento do controle de tráfego junto a pilha de protocolos IP e sua relação com o modelo OSI.

Basicamente, a arquitetura de controle de tráfego permite ao núcleo do sistema processar pacotes que chegam da rede através de uma interface de entrada, para o ponto de ingresso, no qual poderá haver um ou mais métodos de policiamento. O policiamento geralmente descarta pacotes indesejáveis, em função de diversas causas, uma delas por estarem chegando muito rapidamente. Após o policiamento, pacotes são direcionados novamente para a rede, através de uma interface de saída, ou são direcionados para as camadas superiores da pilha de protocolos IP, para processamento posterior. As camadas de níveis superiores também geram dados por elas mesmas e se apóiam nas camadas inferiores para execução de tarefas de encapsulamento, roteamento e eventualmente transmissão, como mostra a Figura 6.2.

O processo de encaminhamento inclui a seleção da interface de saída, a seleção do próximo *hop*, encapsulamento, etc. Quando todas estas tarefas estiverem prontas, os pacotes são enfileirados na interface de saída correspondente, ou ponto de egresso. Portanto, este é um segundo ponto onde o controle de tráfego pode atuar. Controle de tráfego pode, entre outras coisas, decidir se os pacotes deverão ser enfileirados ou descartados, por exemplo, em função do limite da fila ou em função do excesso de tráfego para uma dada taxa de transferência. Também pode decidir sobre a ordem em que os pacotes devam ser transmitidos, de maneira a dar prioridade a certos fluxos de tráfego, pode ainda, atrasar o envio de pacotes, de maneira a limitar o montante do tráfego de saída, etc.

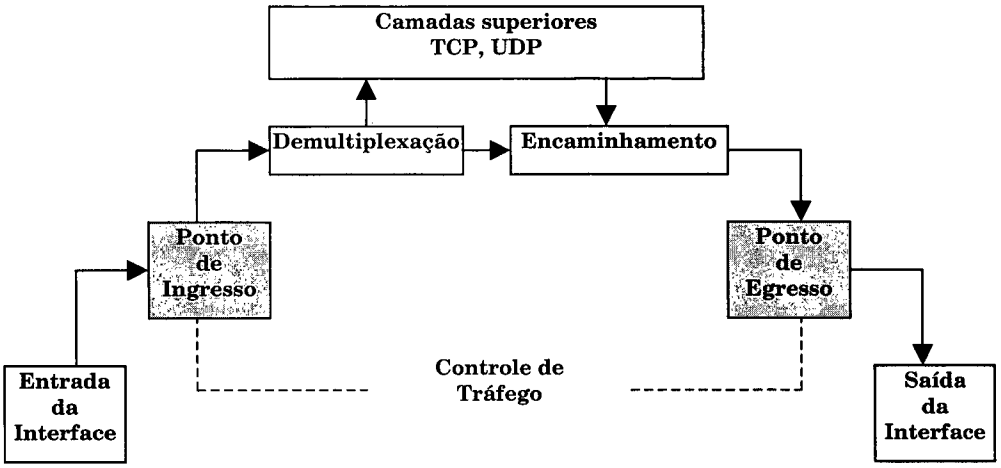

Figura 6.2 Posicionamento da arquitetura de controle de tráfego prevista para a plataforma do sistema operacional Linux.

Uma vez que o controle de tráfego tenha liberado um pacote para transmissão, o controlador de dispositivo de rede o recupera da fila de saída, transmitindo-o para a rede.

6.1.1 Componentes do Controle de Trafego no Linux

O controle de tráfego no núcleo do sistema operacional Linux consiste dos seguintes componentes majoritários: *disciplinas de fila*, *classes*, *filtros*. A Figura 6.3 ilustra uma estrutura de controle de tráfego e de diferenciação de serviços. Deve-se notar que múltiplos filtros podem ser mapeados para uma mesma classe.

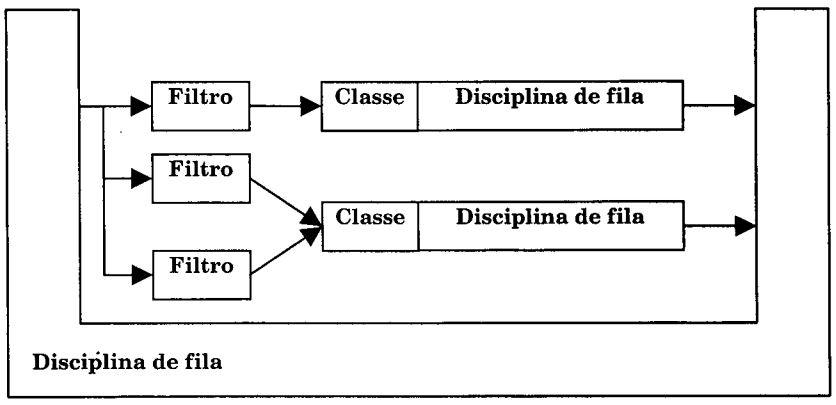

Figura 6.3 Diagrama representativo da relação entre disciplinas de fila, classes e filtros.

Cada dispositivo de rede possui uma disciplina de fila associada, a qual controla o nível de tratamento dado aos pacotes enfileirados neste dispositivo. Por exemplo, **sch_fifo** é uma destas disciplinas, a qual consiste simplesmente de uma única fila FIFO, onde todos os pacotes são armazenados por ordem de chegada, sendo que esta fila será esvaziada tão rápido quanto o dispositivo de rede possa enviar estes pacotes. Disciplinas de filas mais elaboradas podem utilizar filtros, como forma de distinguir entre diversas classes de pacotes, e processar cada classe de forma específica, por exemplo, concedendo a uma determinada classe prioridade sobre as outras.

Disciplinas de filas e classes de serviço estão intimamente relacionadas, devido a que disciplinas de fila, de uma forma genérica, implementam o comportamento esperado de uma classe de serviço. Em contraste a isto, filtros podem ser combinados arbitrariamente com disciplinas de filas e classes, desde que disciplinas de filas façam referência a classes de serviço, para as quais os pacotes devam ser mapeados.

Classes de serviço normalmente não se ocupam em produzir por elas mesmas o armazenamento de pacotes, e sim utilizam disciplinas de filas para isto. Disciplinas de filas podem ser escolhidas arbitrariamente a partir de um conjunto de disciplinas de filas disponíveis. Estas podem fazer referência a classes de serviço, as quais por sua vez, podem fazer referência a disciplinas de filas. Caracterizando, desta forma, um aninhamento recursivo entre disciplinas de filas e classes de serviço.

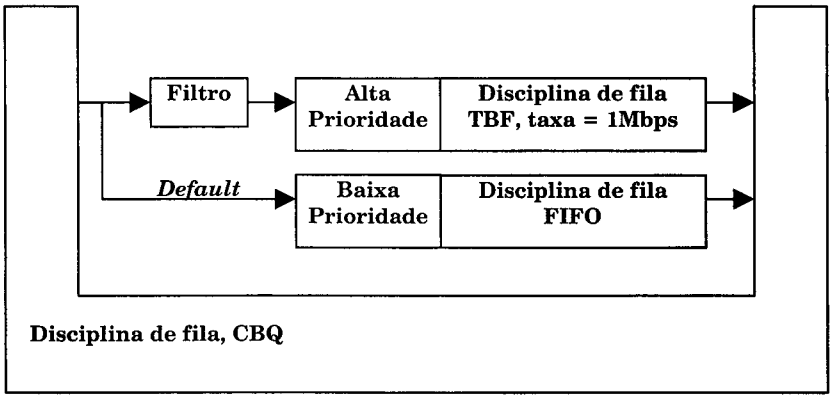

Figura 6.4 Diagrama representativo de uma hierarquia de classes, onde através da utilização de filtros, obtém-se duas classes de serviço, a de *alta* e *baixa* prioridade de tráfego.

A Figura 6.4 mostra um exemplo desta interatividade onde existe uma disciplina de fila com dois níveis de prioridade em função do atraso. Pacotes que forem selecionados pelo filtro serão encaminhados para a classe de prioridade alta, enquanto que todos os outros pacotes irão para a classe de menor prioridade. Enquanto houver pacotes na fila de alta prioridade, estes

são transmitidos antes dos pacotes da fila de menor prioridade, sendo que a disciplina **sch_prio** trabalha desta forma. De maneira a prevenir que a classe de alta prioridade monopolize a plenitude dos recursos, é utilizada uma disciplina de fila que atua como um filtro do tipo *token bucket filter*, ouTBF, o qual assegura uma taxa de no máximo 1 Mbps para esta classe.

Finalmente, o enfileiramento de pacotes de baixa prioridade é feito por uma disciplina de fila do tipo FIFO. Deve-se notar que existem outras maneiras de se conseguir este mesmo resultado, por exemplo, através da adoção da disciplina de fila CQB.

Pacotes são enfileirados quando houver uma chamada para a função de enfileiramento, **enqueu**, de uma disciplina de fila, esta percorre os filtros até que um destes indique um identificador de classe. Uma vez obtido este identificador será feita uma chamada para a função de enfileiramento da disciplina de fila associada à classe identificada. Pacotes que não se enquadrarem a nenhum dos filtros são tipicamente atribuídos a uma classe default.

Tipicamente cada classe de serviço possui uma disciplina de fila associada, mas em principio também é possível que muitas classes de serviço compartilhem a mesma disciplina de fila, ou ainda, uma única disciplina de fila pode ser utilizada por todas as classes da disciplina de fila respectiva.

Geralmente durante a operação de inserção de pacotes, os fluxos correspondentes podem ser policiados, por exemplo, pode ocorrer o descarte de pacote que excederem determinado limite da taxa de transferência.

6.1.1.1 Disciplinas de Fila

Instâncias de disciplinas de fila são identificadas por números de 32 *bits*, os quais são separados em duas partes de 16 *bits*, constituindo um código sob a forma *major:minor*. Para disciplinas de filas o número *minor* é sempre zero, Tentativas de criar uma disciplina de fila com o número *minor* diferente de zero incorre em um erro[15]. Números *major* devem ser únicos por interface, por exemplo, deve existir uma única disciplina de fila rotulada 1:0 na interface eth0, o que não exclui a possibilidade da existência do rótulo 1:0 em outra interface.

Números *major* são associados pelo usuário e devem estar no intervalo 1, 2, 3, ..., 0x7fff. O número *major* de valor zero possui o significado "não-epecificado". Sendo que, números no intervalo de 0x8000 até 0xffff são

[15] Com exceção da disciplina de fila **ingress**, a qual utiliza o número especial ffff:fff1.

atribuídos automaticamente pelo sistema, quando durante a criação de disciplinas que não for especificado seu número *major*.

Números *major* no intervalo de ffff:fff0 até ffff:ffff são reservados, ou possuem significado especial, por exemplo, o número ffff:ffff seleciona a disciplina **egress** que se encontra no topo de todas as disciplinas atribuídas a uma interface. Valores especiais estão definidos em *include/linux/pkt_sched.h* e possuem nomes iniciando por TC_H. Deve-se notar que estes números *major* e *minor* não estão relacionados aos números major e minor utilizados por dispositivos especiais de arquivo.

Quando é criada uma nova disciplina de fila, a mensagem enviada para o núcleo do sistema, **struct tcmsg** descrita em *include/linux/rtnetlink.h*, contém o número de identificação descrito em **tcm_handle** e o ponto de inserção em **tcm_parent**. O valor do ponto de inserção pode ser o número de uma instância previamente criada, ou um dos valores especiais, sendo TC_H_ROOT para a disciplina de topo em **egress**, ou TC_H_INGRESS para a disciplina de fila **ingress**.

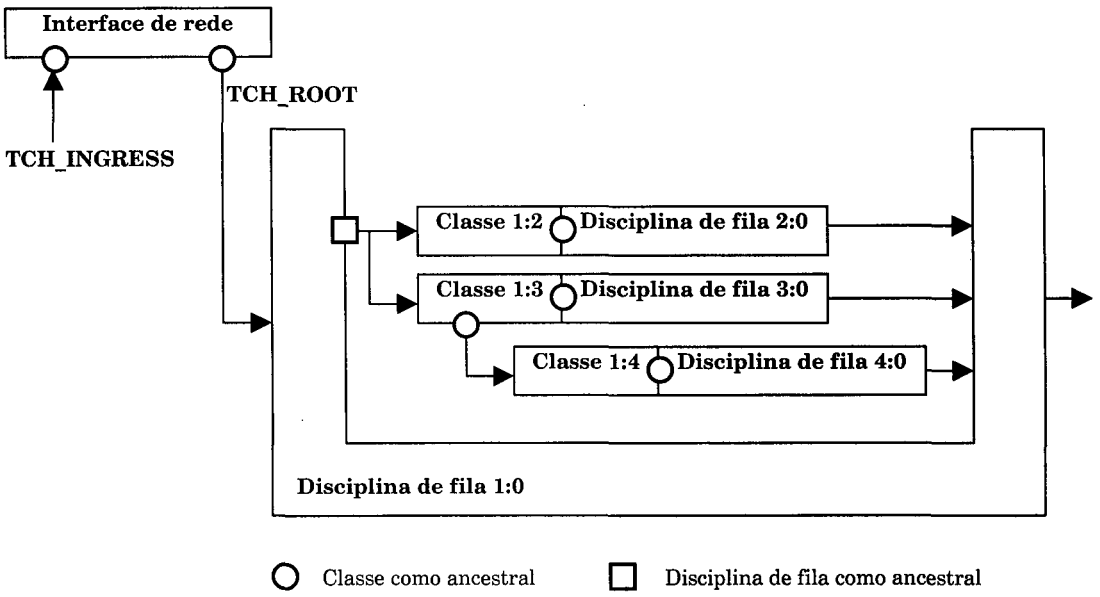

Figura 6.5 Diagrama descritivo do grau de parentescos entre classes e disciplinas de filas. Filtros não foram mostrados por questão de claridade.

A Figura 6.5 ilustra a utilização de graus de parentescos em uma disciplina de fila que possui classes hierárquicas. Por questão de claridade não foram acrescentados filtros. Também deve-se perceber que a numeração da classe começa e dois, pois existe a classe raiz 1:1, a qual não aparece quando utiliza-se este modelo de representação.

6.1.1.2 Funções comuns a disciplinas de fila

Cada disciplina de fila proporciona o seguinte conjunto de funções para controle de sua operação, o que pode ser verificado em **struct Qdisc_ops** em *include/net/pkt_sched.h*:

enqueu Insere um pacote na disciplina de fila. Se a disciplina de fila estiver associada a uma ou mais classes de serviço, então esta função primeiro seleciona uma das classes, para depois invocar a função **enqueue** da disciplina de fila interna correspondente a classe selecionada.

dequeu Retorna o próximo pacote elegível para transmissão. Se a disciplina de fila não contiver nenhum pacote para transmitir é retornado o valor NULL.

requeu Insere um pacote novamente a fila após uma operação de retirada. Esta operação difere de **enqueu** devido a que o pacote deverá ser inserido exatamente do local de que foi removido, e que esta operação não deverá ser incluída na estatística de tráfego cumulativo que perpassa a fila, porque isto já foi feito pela operação **enqueu**.

drop Descarta um pacote da fila.

init Inicia e configura a disciplina de fila.

reset Retorna a disciplina de fila para o seu estado inicial. Todas as filas são esvaziadas, temporizadores são parados, etc. Também, a função **reset** de todas as disciplinas de filas associadas às classes desta disciplina de fila é invocada.

destroy Remove uma disciplina de fila. Remove todas as classes e possivelmente todos os filtros, cancela todos os eventos pendentes e libera todos os recursos mantidos pela disciplina de fila, exceto a estrutura de dados que descreve a própria disciplina de fila em si.

change Modifica a configuração de uma disciplina de fila.

dump Retorna os dados para diagnóstico e manutenção. Tipicamente, esta função retorna todas as configurações importantes de uma disciplina de fila e suas variáveis de estado.

Todas estas funções geralmente são acessíveis, através de um ponteiro para uma estrutura, **struct Qdisc**, o qual deverá corresponder a disciplina que se quer fazer referência.

Quando uma instância de uma disciplina de fila for criada ou modificada, um vetor contendo opções de criação, do tipo **struct rtattr*** declarado em *include/linux/rtnetlink*, é passado para a função **init**. Cada opção é codificada em função de seu tipo, comprimento e pelo próprio valor de seu argumento, ou seja, zero ou mais bytes de dados.

Tipos de opções e estruturas de dados utilizadas para seus valores estão declaradas em *include/linux/pkt_sched.h*. O vetor de opções é submetido a um *parser* através da chamada a função **rtattr_parse**, a qual retorna uma matriz de ponteiros para os elementos individuais, indexados pelo tipo da opção. O comprimento e conteúdo de uma opção pode ser acessado através das macros RTA_ PAYLOAD e RTA_DATA, respectivamente. Vetores de opções são passados entre o espaço de programa do usuário e o núcleo do sistema através da utilização do mecanismo *rtnetlink*.

6.1.1.3 Classes

Classes de serviço podem ser identificadas de duas maneiras: primeiro pelo identificador da classe, ou simplesmente ID da classe[16], o qual é atribuído pelo usuário; segundo por seu identificador interno, ou ID interno, o qual é atribuído pela disciplina de fila. Este último deve ser único em uma dada disciplina de fila e pode ser um índice, um ponteiro, etc. Deve-se notar que o valor zero possui o significado especial "não-encontrado", sendo um dos valores de retorno possíveis da função **get**. O ID da classe é do tipo **u32**, enquanto que o ID interno é do tipo *unsigned long*. Internamente ao núcleo do sistema operacional, faz-se referência a uma classe pela utilização de seu ID interno. Somente as funções **get** e **change** utilizam o ID da classe, na primeira forma.

Deve-se perceber que múltiplos identificadores de classes podem ser mapeados para um mesmo identificador interno de classe. Neste caso, o ID da classe, pode ser um valor resultante de uma regra de classificação utilizada pelo mecanismo classificador, o qual a disciplina de fila ou a própria classe poderá utilizar.

Identificadores de classes são estruturados da mesma forma que identificadores de disciplinas de fila, sendo que seu número *major* indica a instância da disciplina de fila para qual a classe está associada, e seu número *minor* identifica uma entre muitas classes possíveis desta mesma instância. O número *minor* pode estar no intervalo 0,1,2,3, ..., 0xffff.

---

[16] O identificador da classe, ou ID da classe, atribuído pelo usuário é a visão externa que o usuário tem da associação de classes com disciplinas de fila. Internamente ao núcleo do sistema este identificador possui outra representação, por exemplo, um ponteiro para uma classe, portanto, designa-se ID interno da classe.

Identificadores de classe e identificadores internos de classe são únicos por disciplinas de fila. Sendo que a identificação de uma classe também contém o número da disciplina de fila associada implica que estas também deverão ser são únicas por interface.

6.1.1.4 Funções comuns a classes

Disciplinas de fila que suportam o conceito de classes, contém o seguinte conjunto de funções as quais permitem a manipulação destas classes, definidas na estrutura **struc Qdisc_class_ops** em *include/net/pkt_sched.h*:

graft Anexa uma nova disciplina de fila a uma classe e retorna a disciplina de fila previamente utilizada.

leaf Retorna a disciplina de fila de uma classe.

get Procura por uma classe, utilizando seu identificador de classe, e retorna seu identificador interno. Se a classe mantiver um contador de referência, então **get** irá incrementa-lo.

put Esta função é chamada sempre que uma classe que for previamente referenciada através de **get**, sendo que deve ser decrementado seu número de referências. Se a classe mantiver um contador de referências, **put** irá decrementá-lo. Se a contagem atingir o valor zero, **put** irá remover a classe.

change Modifica as propriedades de uma classe. Também é utilizada para criar novas classes, quando aplicável, pois algumas disciplinas de fila possuem um número constante de classes, as quais são criadas no momento de iniciação da disciplina.

delete Ocupa-se em atender a solicitações de retirada de uma classe. Primeiro é verificado se a classe não está sendo utilizada por nenhum dos filtros. Neste caso, os filtros são desativados e a classe é removida.

walk Percorre todas as classes de uma disciplina de fila e invoca a função *callback* de cada uma delas. Isto é feito para obtenção de dados para diagnóstico de todas as classes de uma disciplina de fila.

tcf_chain Retorna um ponteiro para a âncora da lista de filtros associados à classe. Este ponteiro é utilizado para manipular a lista de filtros.

bind_tcf Informa a disciplina de fila que uma instância de um elemento de filtro está para ser associado a uma de suas classes. Esta função é utilizada de maneira análoga a função **get**, exceto quando a disciplina de fila necessite explicitamente refugar a remoção de uma de suas classes, enquanto houverem filtros associados a elas de maneira a manter a integridade referencial.

unbind_tcf Remove uma instância de um elemento de filtro, de uma classe. Esta função é análoga a função **put**.

dump_class Retorna dados para diagnóstico de classes, da mesma maneira que a função **dump** atua sobre disciplinas de fila.

O vetor de opções passados para a função **change** possui a mesma estrutura do vetor passado para a função **init** da disciplina de fila. As declarações correspondentes localizam-se em *include/linux/pkt_sched.h*.

6.1.1.5 Filtros

Filtros são mantidos em listas de filtros, os quais por sua vez podem ser mantidos por disciplinas de filas ou por classes de serviço, dependendo do projeto concebido para a disciplina de fila. Listas de filtros são ordenados por ordem de prioridade ascendente. Ainda, cada entrada na lista é codificada em função do protocolo ao qual o filtro se aplica. Estes números de protocolos são atribuídos para **skb->protocol** e estão definidos em *include/linux/ if_ether.h*. Filtros para o mesmo protocolo, na mesma lista de filtros, devem possuir prioridades diferentes.

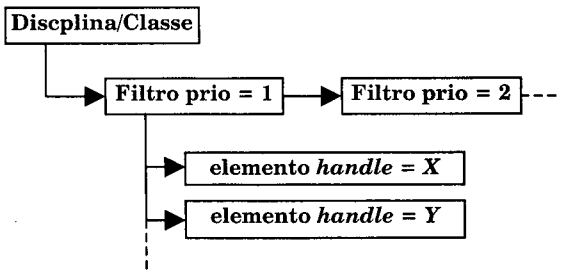

Figura 6.6 Descrição da estrutura de dados que possa ser utilizada para a associação de filtros e elementos de filtros junto a disciplinas de fila ou classes.

A estrutura de um filtro pode permitir a existência e controle de elementos de filtros internos. Filtros são referenciados por *handles* de 32 *bits*. Estes *handles* são similares aos identificadores de classes, sendo que estes não são

separados em números *major* e *minor*. O *handle* de valor zero sempre refere-se ao próprio filtro. Como para classes, filtros também possuem identificadores internos, os quais são obtidos através da função **get**. Identificadores de filtros internos são representados pelo tipo de dados *unsigned long*, sendo que o valor zero indica "não encontrado". *Handles* e identificadores internos de filtros são únicos por instância de cada filtro. A organização interna de um filtro pode ser arbitrária. A Figura 6.6 mostra um filtro com uma lista de elementos internos.

Deve ser observado que, diferente de classes, filtros ou seus elementos não possuem "filtros de seus próprios identificadores" que permitam apontar diretamente para um filtro específico ou para um de seus elementos em uma interface. Ao invés disto, estes são identificados pela disciplina de fila ou classe para as quais foram registrados, quer pelo protocolo como por seus níveis de prioridade e para o caso de elementos de filtros através de seus *handles*.

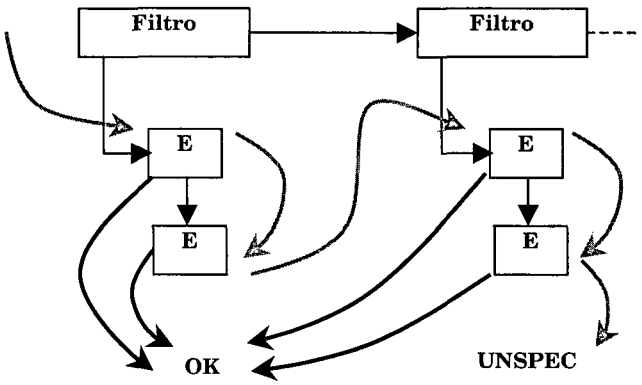

Figura 6.7 Demonstração do fluxo de busca por um elemento de filtro cuja chave de descrição esteja relacionada. Caso não exista é retornado o valor UNSPEC.

A Figura 6.7 mostra a ordem na qual filtros e seus elementos podem ser examinados. Uma lista encadeada a qual é processada seqüencialmente, é uma das possíveis maneiras de representação interna de um filtro. De outra forma, poderia ser utilizado uma tabela *hash*, uma árvore, ou outras estruturas que façam sentido para o filtro específico.

6.1.1.6 Funções comuns aos filtros

Filtros são controlados através das seguintes funções, definidas na estrutura **struct tcf_proto_ops** em *include/net/pkt_cls.h*:

classify Executa a classificação e retorna valores na forma

TC_POLICE_. Se o resultado não for TC_POLICE_ UNSPEC, então será retornado o identificador de classe, opcionalmente será retornado o ID interno da classe contido na estrutura **struct tcf_result** apontada por **res**. Se o ID interno da classe for omitido, o valor zero deverá ser atribuído para **res->class**.

init Inicia o filtro.

destroy Esta função é chamada para remover um filtro. Também as disciplinas de fila tais como a **sch_atm**, **sch_cbq**, **sch_dsmark** e **sch_ingress** utilizam esta função para remover filtros no momento da remoção da uma classe. Se um filtro ou um de seus elementos estiver registrado junto a uma classe, estes registros serão cancelados através da chamada da função **unbind_tcf**.

get Procura pelo elemento de um filtro através de seu *handler* e retorna o ID interno do filtro.

put Esta função é chamada quando um elemento de um filtro foi previamente referenciado pela função **get** e não está mais sendo utilizado.

change Configura um filtro novo ou modifica as propriedades de um filtro existente. Parâmetros de configuração são passados através dos mesmos mecanismos que são utilizados para disciplinas de filas e classes. Esta função registra a inclusão de um novo filtro ou de um de seus elementos através de uma chamada para **binf_tcf**.

delete Elimina um elemento ou um filtro. Para eliminar um filtro por inteiro, deve ser utilizada a função **destroy**. Esta distinção é transparente para o usuário e é feita através de *net/sched/cls_api:tc_ctl_tfilter*. Se o elemento de um filtro foi registrado em uma classe, este registro é cancelado pela chamada da função **unbind_tcf**.

walk Percorre todos os elementos de um filtro e provoca uma chamada para a função *callback* de cada um destes elementos. Isto é feito para obter-se dados para diagnóstico.

dump retorna dados para diagnóstico de um filtro ou de um de seus elementos.

6.2 Plataforma de Teste

A plataforma de teste será constituída pela estrutura atual do provedor tomando-se por base os seguintes requisitos:

1) O ambiente do provedor irá constituir um ambiente real sobre o qual deverão ser localizadas classes de serviços, em função de uma configuração atual e futura.

2) O nodo de borda será constituído por um computador, no qual será instalado o sistema operacional Linux, conjuntamente as ferramentas de software para o controle e gerenciamento de tráfego.

3) Os itens anteriores deverão ser suficientes para permitir a seleção de um algoritmo de diferenciação de serviços, ou de compartilhamento de banda, a ser aplicado junto ao nodo de borda.

6.2.1 Ambiente do provedor

A topologia da rede do provedor é dada pela Figura 6.8. Nela pode-se verificar prontamente a existência de três tipos de tráfego concorrentes: O primeiro provocado pelos servidores, o segundo pelos clientes dedicados Frame Relay e o terceiro pelos clientes discados.

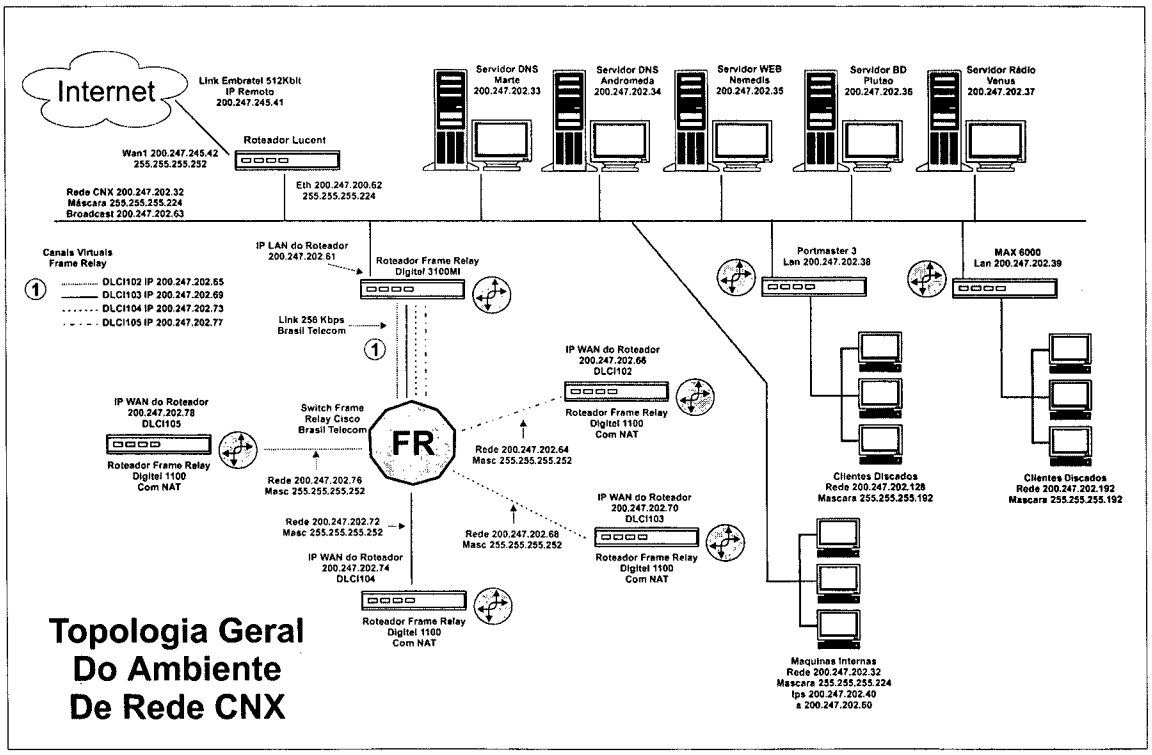

Figura 6.8 Ambiente do provedor e as possíveis categorias de serviços, baseados na aplicação, usuário e organização.

O conjunto dos servidores constitui a categoria de clientes na forma de aplicações, entre elas estão os servidores para os protocolos http, ftp, smtp e

rtp. Os clientes dedicados recaem para a categoria de clientes do tipo organização. Enquanto que os clientes discados localizam-se na categoria usuário final.

6.2.2 Classes de tráfego

A identificação dos agregados são obtidos diretamente das sub-redes formadas a partir da divisão da classe C de números IPs atribuída ao provedor. Neste caso foram utilizados números IPs de classes não válidas para ilustração do problema. Sendo que ao provedor foi atribuída a classe C 192.168.168.0. Esta por sua vez tem os seus primeiros 64 endereços IPs reservados a sub-rede de servidores. Enquanto que aos clientes dedicados Frame Relay foram reservados 64 endereços IPs no intervalo de 64 a 128. Os 128 endereços IPs restantes são atribuídos aos clientes discados. Ainda, a rede do provedor está ligada a rede da operadora através da sub-rede 172.16.16.64/30, a qual é constituída por quatro endereços IPs, um para a rede, outro para *broadcast* e os dois restantes para as interfaces WAN de cada ponto do enlace, como mostra a Figura 6.9.

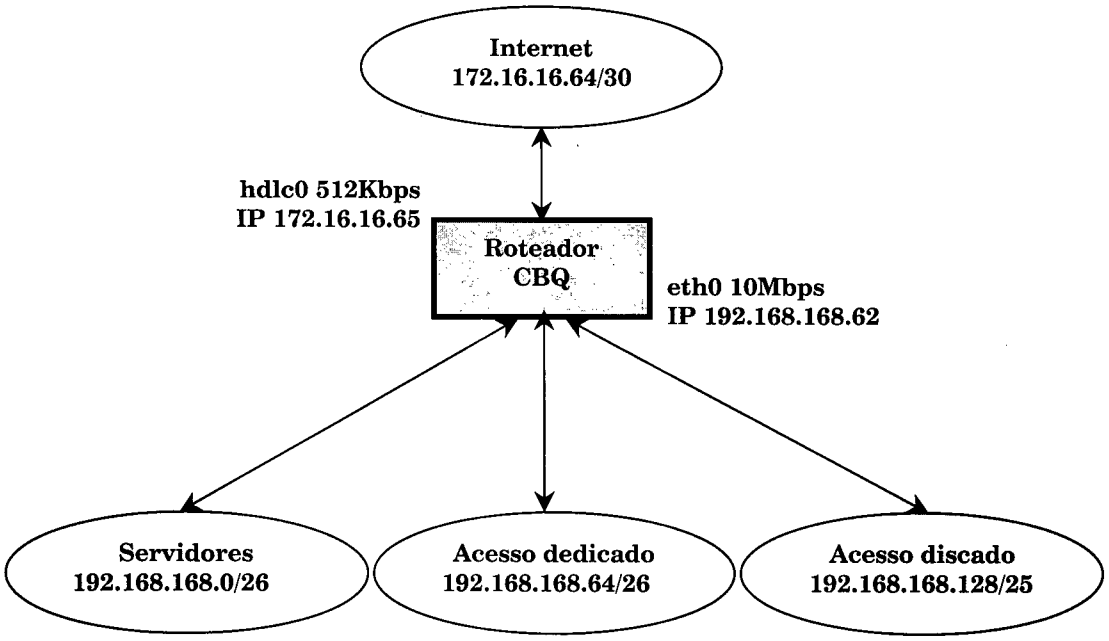

Figura 6.9 Descrição dos agregados de fluxos de tráfego presente na estrutura do provedor.

6.2.2.1 Níveis de agregação

Naturalmente seria possível a criação de três classes de serviço, como sugere a Figura 6.9, mas devido à discrepância de velocidade entre os *links* optou-se por considerar o acesso discado e dedicado como uma única classe de serviço, para efeitos de teste e de tomada de dados. Permanecendo, desta forma, duas classes de serviços: uma caracterizada pelo conjunto de servidores, sob a sigla CSS, e outra caracterizada pelos usuários de acesso dedicado e discado, sob a sigla CSU.

De forma a evitar que uma destas classes de serviço monopolize os recursos de banda disponível em tempos de congestionamento, a capacidade total do canal é dividida em parcelas iguais entre estas duas classes. A Figura 6.10 mostra a estrutura hierárquica de compartilhamento.

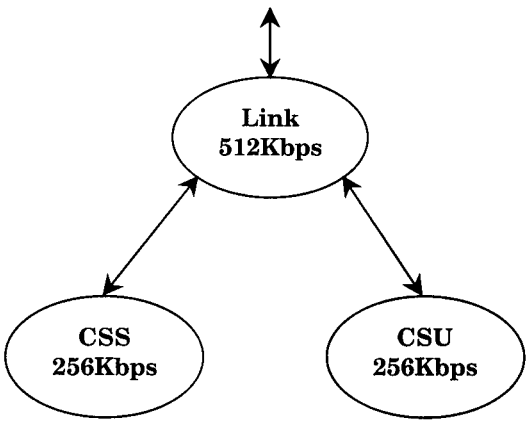

Figura 6.10 Hierarquia de classes de serviços. Classe de serviços dos servidores (CSS) e classe de serviço aos usuários (CSU).

Neste cenário, procura-se garantir aos clientes internos a rede do provedor um nível de desempenho mínimo de 256Kbps para acesso ao *link* Internet, quando o nível de recursos tenderem para o seu limite. Da mesma forma, pretende-se assegurar um nível de desempenho mínimo de 256Kbps aos clientes que procuram por serviços Internet disponíveis nos servidores internos ao provedor.

6.2.3 O roteador CQB

O roteador utilizado consiste em uma plataforma baseada em um computador Pentium 233MHz, no qual foi instalado o sistema operacional Linux Red Hat 7.1. Este roteador possui além de uma interface Ethernet de 10Mbps a qual constitui a LAN do roteador, uma interface WAN de 2Mbps do tipo PC300 da Cyclades. Esta última também requer, como qualquer outro dispositivo, que

seu *device driver* seja carregado para o sistema e compilado, para então estar disponível como uma interface de rede. Neste caso, esta interface está operando com um protocolo de enlace do tipo PPP com um *link* de rádio, de 512Kbps, junto a Embratel.

6.2.3.1 Configuração do roteador

Estando o conjunto de ferramentas de software e dispositivos de hardware devidamente configurados, pode-se então passar a utilizar tais recursos para a devida função, que neste caso, corresponde à aplicação de uma hierarquia de controle de tráfego sobre as interfaces *eth0* e *hdlc0*, como descreve a Figura 6.11.

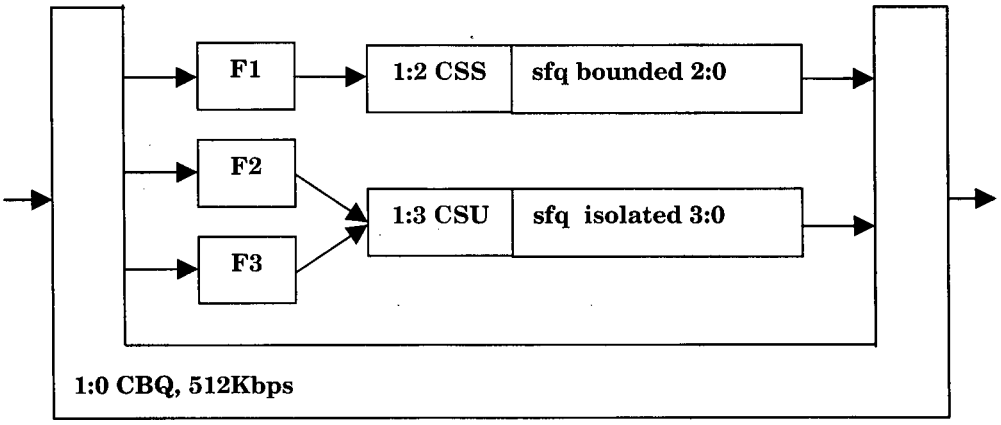

Figura 6.11 Descrição do diagrama hierárquico de classes e disciplinas de filas que serão aplicados as interfaces do roteador CBQ.

A disciplina de fila raiz é do tipo **cbq**, sendo o filtro F1 responsável em redirecionar o fluxo de pacotes gerado pelos servidores para a classe de serviço CSS. Os filtros F2 e F3, da mesma forma, se ocupam em redirecionar o fluxo de pacotes originado pelos clientes, de acesso dedicado e discado, para a classe de serviço CSU.

A cada classe de serviço está associada uma disciplina de fila do tipo **sfq**, *stochastic fair queueing*. A opção por esta disciplina de fila junto às classes folhas deve-se ao fato de que esta disciplina de fila foi concebida exatamente para operar sobre agregados construídos com base nos endereços de origem e destino dos pacotes.

A mesma estrutura hierárquica será aplicada, trocando-se apenas a disciplina raiz CBQ pela disciplina HTB, para efeitos de tomadas de dados e de comparações de desempenho.

6.2.3.2 Configuração da interface *eth0*

No apêndice A são dados os scripts de configuração da interface *eth0,* os quais efetivam, através da utilização da ferramenta de software **tc** de configuração do controle de tráfego, a criação de uma hierarquia de compartilhamento de banda do tipo **cbq** sobre a interface *eth0.*

6.2.3.3 Configuração da interface *hdlc0*

Da mesma forma que para a interface *eth0*, no apêndice A são dados os scripts de instalação da estrutura hierárquica de compartilhamento de banda junto à interface *hdlc0.*

6.3 Análise da Interface eth0

Sobre esta interface será feito o controle sobre o fluxo de pacotes provenientes da Internet e que tem como destino as classes CSS e CSU da hierarquia de controle. Este controle tem como objetivo evitar que uma destas classes monopolize os recursos disponíveis de acesso a Internet.

6.3.1 Cenário 1

Neste cenário foram feitas tomadas de dados fazendo-se com que as classes da hierarquia de controle fossem restringidas pelas propriedades dispostas pela Tabela 6.1.

Classes	Isolada	Limitada
Link	X	
CSS		X
CSU		X

Tabela 6.1 Configuração das classes de serviço para o cenário 1.

Neste cenário o comportamento esperado para as classes CSS e CSU é que ambas se mantenham restritas as suas cotas de compartilhamento, não importando o nível de banda excedente que possa existir em um determinado intervalo de tempo.

6.3.1.1 Método CBQ

O comportamento geral demonstrado pelo método CBQ, neste cenário, evidencia um dos requisitos importantes de todo o objetivo de transformar o modelo Internet em uma rede multi-serviços: o comportamento de uma classe não deve ser influenciado pelo comportamento de outra classe. Como pode ser observado na Figura 6.12.

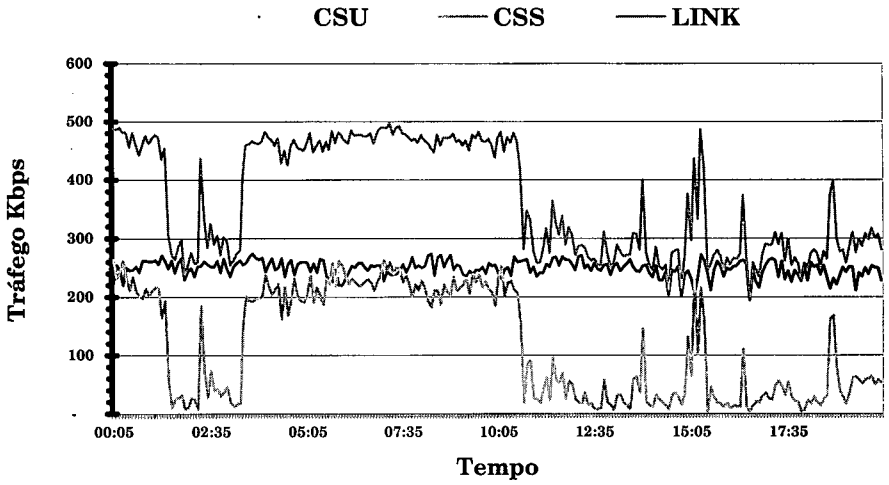

Figura 6.12 Tomada de dados das classes Link, CSS e CSU usando o método CBQ. Sendo a classe Link isolada e as classes CSS e CSU limitadas.

Nesta figura a classe raiz ou Link, de cor cinza, ora tende ao seu limite de 512Kbps ora apresenta recursos disponíveis. A classe CSU, em negrito, limitada a sua cota de compartilhamento de 256Kbps, por todo período flutuou em torno de sua capacidade máxima. Mesmo para o períodos que havia recursos disponíveis esta não recebeu o excedente disponível, devido ao fato desta estar limitada. A classe CSS, cinza claro, em seu período mais ativo tendeu a permanecer a sua cota de compartilhamento, liberando recursos em seguida.

6.3.1.2 Método HTB

A tomada de dados registrada durante a aplicação do método HTB, para este cenário, demonstra uma alta taxa de ocupação do canal, como ilustra a Figura 6.13.

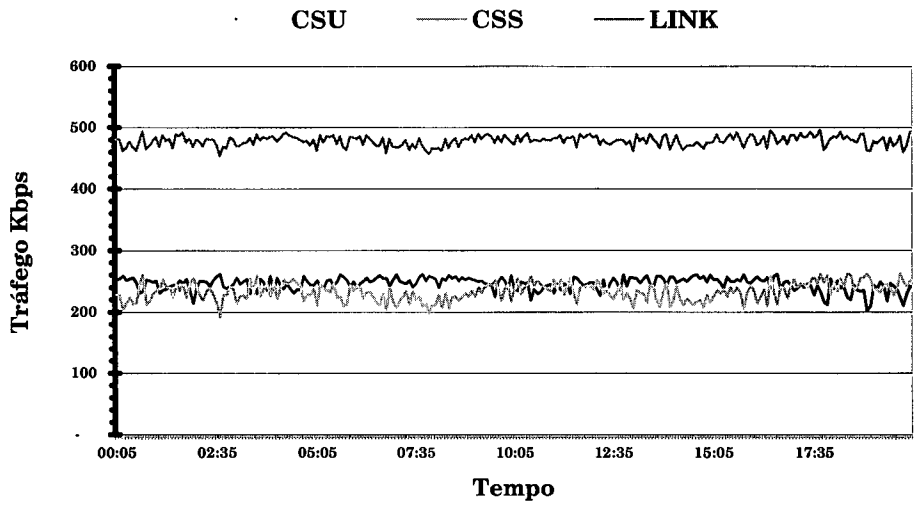

Figura 6.13 Tomada de dados das classe Link, CSS e CSU usando o método HTB. Sendo permitido as classes CSS e CSU utilizar no máximo 256Kbps cada uma.

Nesta figura pode se perceber que este método também condiciona as classes as suas cotas de compartilhamento, com pequenas flutuações que podem ser produzidas pela variação da ocupação do canal e da presença de buffers nos controladores de dispositivo.

6.3.2 Cenário 2

Neste cenário foram feitas tomadas de dados fazendo-se com que as classes da hierarquia de controle fossem restringidas pelas propriedades dadas na Tabela 6.2.

Classes	Isolada	Limitada
Link	X	
CSS		X
CSU	X	

Tabela 6.2 Configuração das classes de serviço para o cenário 2.

Neste cenário o comportamento esperado para a classe CSS é que esta permaneça restringida a sua cota de compartilhamento e não receba nenhum percentual de banda excedente que possa ocorrer em algum intervalo de tempo. Espera-se que a classe CSU, receba além de sua cota de

compartilhamento quando houver excedente de recursos disponível e que no mínimo seja respeita a sua parcela de banda.

6.3.2.1 Método CBQ

A tomada de dados ilustrada pela Figura 6.14, registra o comportamento esperado deste método para esta configuração.

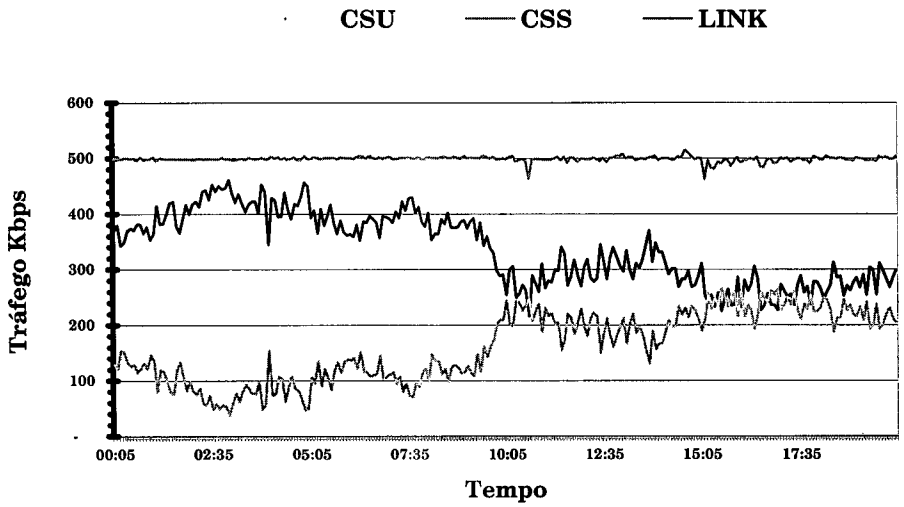

Figura 6.14 Tomada de dados das classe Link, CSS e CSU usando o método CBQ. Sendo a classe CSS limitada e a classe CSU isolada.

Nesta figura pode-se observar que a classe CSS, cinza claro, tende a estar limitada a sua cota compartilhamento. A classe CSU, sendo do tipo isolada, recebe toda a cota excedente de recursos que estiver disponível, enquanto a classe CSS nada recebe além de sua própria cota. Em todo o período pode-se observar a concorrência pelos recursos excedentes provocados flutuação da classe CSS.

Este comportamento caracteriza o segundo maior objetivo deste método: o excedente de recursos disponível deve ser distribuído entre as classes ativas em proporção as suas cotas de compartilhamento respectivas.

6.3.2.2 Método HTB

A tomada de dados para o método HTB, para este cenário, registra um comportamento conforme esperado, como ilustra a Figura 6.15.

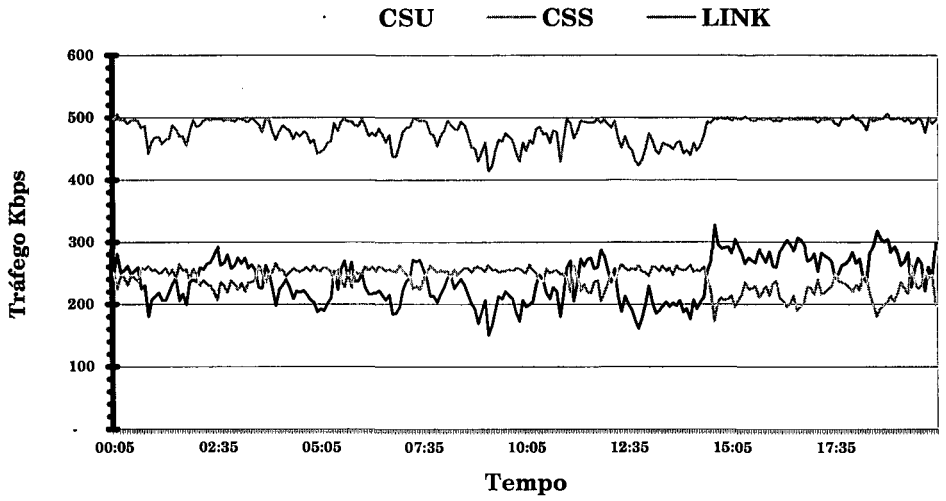

Figura 6.15 Tomada de dados das classe Link, CSS e CSU usando o método HTB. Sendo permitido a classes CSS utilizar no máximo 256Kbps e a classe CSU poderá receber o excedente disponível além de sua cota de 256Kbps.

Nesta figura pode-se perceber que a classe CSS, cinza claro, no início do período manteve-se dentro de sua cota de compartilhamento, mesmo quando havia recursos disponíveis, o que caracteriza o comportamento esperado para esta classe. No final do período a classe CSU, negrito, recebeu o excedente produzido pela liberação de recursos da classe CSS, o que também caracteriza o comportamento esperado.

6.4 Análise da Interface hdlc0

A aplicação da hierarquia de controle, ilustrada pela Figura 6.11, sobre esta interface tem como objetivo condicionar o tráfego de pacotes originados pela rede interna do provedor e que tem como destino a saída desta interface, ou seja, a Internet. Desta forma, procurou-se equalizar os fluxos de pacotes de entrada e saída entre as interfaces eth0 e hdlc0, dado as discrepâncias de capacidade destas interfaces.

Dado que o ambiente de teste foi produzido para uma situação real, este também demonstrou o comportamento padrão verificado nas interfaces entre redes privativas e a Internet: o fluxo de pacotes é muito maior no sentido da Internet para a rede privativa, como verificado comparando-se as análises das interfaces etho e hdlc0.

Portanto, como neste caso não se tem controle da carga imposta a rede no sentido rede privativa para a Internet, ficaram prejudicadas as análises desta interface. Embora a funcionalidade de seu propósito ainda permaneça, ou seja, a equalização da capacidade entre as interfaces.

Mesmo assim, serão apresentados diagramas de tomadas de dados para esta interface independente de cenários.

6.4.1 Método CBQ

A tomada de dados ilustrada pela Figura 6.16, registra o compartilhamento do canal de saída, no sentido rede privativa para a Internet, entre os fluxos de pacotes produzidos pelas classes CSS e CSU.

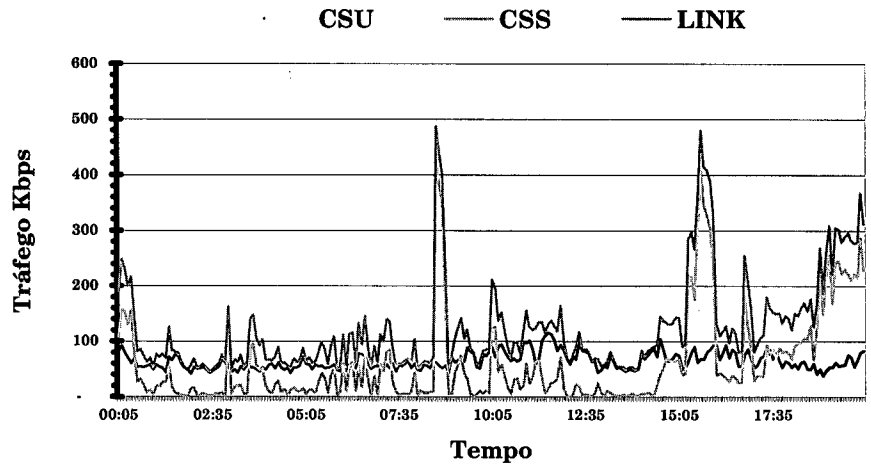

Figura 6.16 Tomada de dados das classe Link, CSS e CSU usando o método CBQ. Sendo permitido as classes CSS e CSU concorrem pelo excedente dos recursos disponíveis ale de suas cotas de 256Kbps.

Nesta figura pode-se verificar que a classe Link, cinza, acompanha a demanda da classe CSS, cinza claro. Este tráfego representa aquele originado pelas aplicações instaladas nos servidores http, ftp, smtp. Enquanto que o tráfego produzido pelos clientes, neste sentido, deve-se em sua maioria pelos pacotes de controle ICMP gerados pela pilha de protocolos IP.

6.4.2 Método HTB

A tomada de dados ilustrada pela Figura 6.17, registra o compartilhamento entre as classes CSS e CSU pelos recursos disponíveis sobre a interface hdlc0, no sentido rede privativa para a Internet.

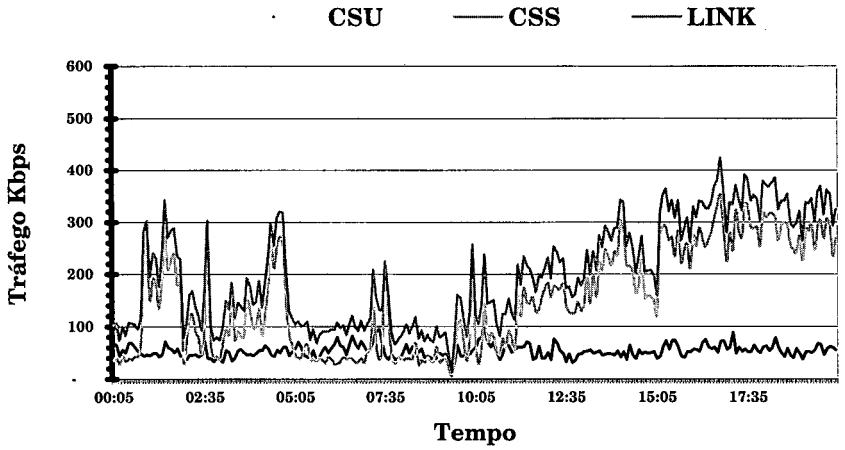

Figura 6.17 Tomada de dados das classe Link, CSS e CSU usando o método HTB. Sendo permitido as classes CSS e CSU concorrem pelo excedente de recursos além de suas cotas de compartilhamento de 256Kbps.

Nesta figura pode ser observado que a classe Link, cinza, acompanha a demanda da classe CSS, cinza claro. Como no caso anterior, fica evidente que neste sentido de tráfego ocorre uma demanda maior provocada pelas aplicações instaladas nos servidores. Enquanto que o volume de tráfego produzido pelos clientes se mantém praticamente o mesmo do caso anterior.

6.5 Análise dos Resultados Obtidos

O objetivo maior da utilização destes mecanismos de diferenciação de serviços é o de concretizar todos os conceitos desenvolvidos na investigação sobre a complexidade de transformação do modelo Internet em uma rede multi-serviços, bem como, devido a sua aplicabilidade. Pois, internamente estes métodos utilizam algoritmos de escalonamento baseado no método DRR, *Déficit Round Robin*, o qual pode ser substituído por soluções mais eficientes e atuais.

Ambos os métodos demonstraram os comportamentos esperados na especificação de cada cenário de teste.

O método CBQ demonstra uma capacidade de isolamento de uma classe de serviço do comportamento de outras classes, menor do que o método HTB. Isto se deve em função do método CBQ utilizar um mecanismo de medição da taxa de ocupação do canal pela classe de serviço de desempenho inferior ao do método HTB.

O método HTB pelo fato de utilizar dois mecanismos de token bucket, um para determinar a cota de compartilhamento do canal e outro para determinar a quantidade de empréstimo do excedente de banda, permite um controle maior sobre o isolamento de cada classe de serviço de outra. Pois, se os parâmetros de ambos os medidores forem iguais então a classe é limitada a sua cota de compartilhamento. O valor atribuído ao segundo token bucket determina a quantidade que uma classe possa emprestar, ou receber do excedente de banda. Este controle fino, do ponto de vista da formação de preços apresenta maior imparcialidade pois as regras de diferenciação entre serviços são mais claras e consistentes.

Para o caso de uma aplicação real, o método HTB seria o método selecionado. No entanto, este ainda está em desenvolvimento e uma formalização mais adequada faz-se necessário.

6.6 Resumo

Neste capítulo foi verificado o comportamento do modelo IP quando a este foram submetidos os métodos CBQ e HTB de compartilhamento do canal de comunicação com a Internet entre as classes CSS e CSU, para diferentes formas de interação entre estas classes.

Das observações pode então ser verificado que é possível condicionar o comportamento do modelo IP de forma a produzir sobre um mesmo canal físico, sub-canais distintos, os quais podem ser caracterizados na forma de classes de serviços.

Também foi verificado que podem ser impostas restrições sobre estas classes de forma que estas se mantenham limitadas a sua cota de compartilhamento independente do comportamento de outras classes.

Também pode ser imposto regras de compartilhamento do excedente de recursos entre classes concorrentes. Esta possibilidade caracteriza um grande diferencial para a concepção de redes multi-serviço, pois a reserva de recursos

ocorre somente quando for necessária. Não sendo, então sua parcela de banda é imediatamente distribuída entre as outras classes.

Capítulo 7

Conclusão

Os processos de investigação e de experimentação desenvolvidos neste trabalho foram conduzidos de forma que permitissem explorar os graus de dificuldades inerentes ao processo de transformação do modelo Internet em uma rede multi-serviço.

Transformar o modelo Internet em uma rede *gerenciável*, *atrativa* e *lucrativa* tem como questão fundamental à verificação de que sobre esta rede possam ser construídos "produtos com características distintas". Sendo possível, então pode-se fazer uma analogia que um domínio de rede será semelhante a uma linha de produção flexível, a qual tem como produto, *serviços de rede*.

Neste trabalho foi identificado que serviços de rede possuem três níveis de abstração associados: Classes de Serviços ao Cliente, Classes Abstratas de Serviços de Rede e Mecanismos de Rede.

- O nível de abstração, **Classes de Serviços ao Cliente**, permite capturar a visão do cliente sobre serviços de rede e a conseqüente determinação de seus requisitos e características inerentes, as quais geralmente são expressas em termos *quantitativos*, *qualitativos* e *relativos*. Também, o seu isolamento, permite que campanhas de marketing façam uso deste nível de abstração durante a divulgação do serviço referindo-se a um produto com características bem definidas e atrativas.

- O nível de abstração, **Classes Abstratas de Serviço de Rede**, é na verdade um encapsulamento da semântica definida para o termo PHB previsto pelo modelo de Serviços Diferenciados. O seu isolamento permite

capturar os requisitos e características intrínsecas da classe de serviços ao cliente de maneira a determinar o comportamento previsto para esta classe, geralmente determinado pelos parâmetros de *atraso*, *variação de atraso*, *perda de pacotes* e *prioridades relativas* entre classes de serviço. Ainda, pelo fato de ser abstrata é independente do mecanismo que deverá implementar este comportamento.

- O nível de abstração, **Mecanismo de Rede**, é na verdade a implementação do comportamento esperado para uma classe abstrata de serviços de rede. Sendo que esta ultima parametriza, através dos requisitos e características intrínsecas capturadas da classe de serviço ao cliente, a evolução ou o comportamento destes mecanismos.

A constatação destes três níveis de abstração e o seu isolamento permite que o termo *diferenciar* seja aplicado a cada uma destas instâncias com objetivos bem definidos: diferenciar a percepção do cliente sobre serviços de rede; diferenciar entre classes de serviços de rede e prover mecanismos para a diferenciação de serviços.

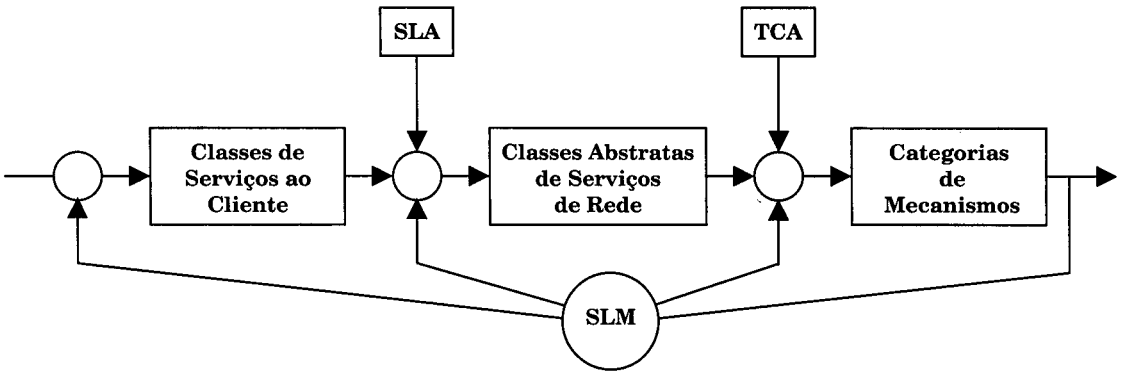

Figura 7.1 Relacionamentos entre os níveis de abstração de serviços de rede e os seus níveis de gerenciamento.

As inter-relações entre estes níveis de abstração são efetuadas por relações contratuais. Sendo que um acordo de nível de serviço, SLA, captura os requisitos do cliente e forma a base para o projeto de classes de serviços de rede. Uma vez determinadas, classes de serviços de rede expõem o seu comportamento através de um acordo de controle de tráfego, ou TCA. Mecanismos de rede, por sua vez, são governados pelos parâmetros descritos em um TCA de forma a exibir o comportamento esperado para a classe de serviço. Por fim, cada um destes níveis exige formas de gerenciamento diferenciadas, SLM, de maneira garantir a consistência e imparcialidade classes de serviços.

Do ponto de vista da rede todos estes níveis de abstração serão em sua ultima instância traduzidos para uma hierarquia de controle de tráfego, a qual será associada a uma interface de um nodo da rede. Onde geralmente filtros

capturam e policiam o perfil do comportamento esperado pelo cliente. Classes parametrizam a evolução de disciplinas de filas. Todos estes elementos postos em conjunto determinam mecanismos de diferenciação de serviços de rede.

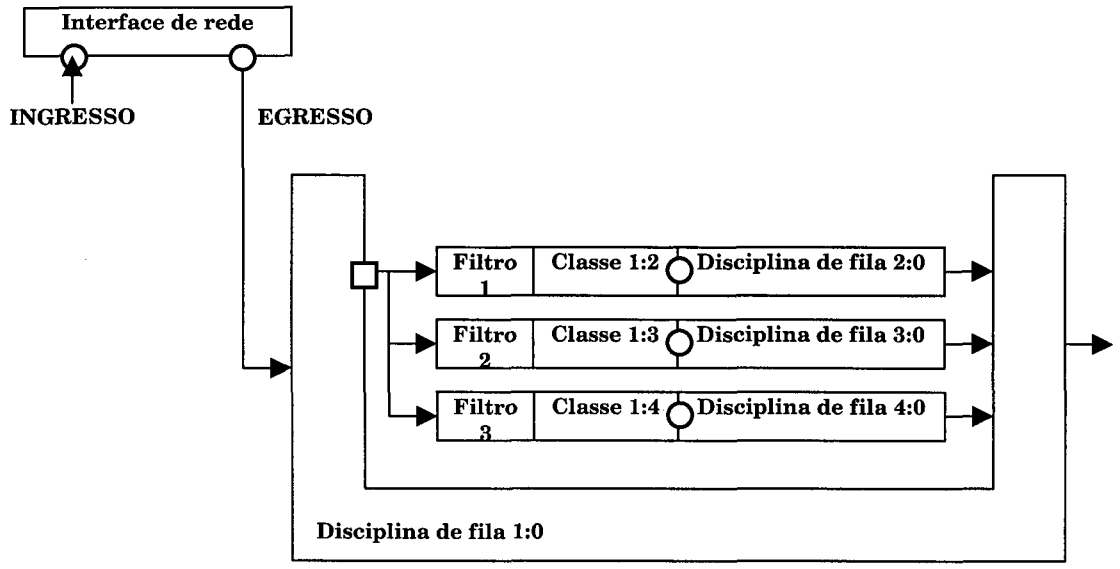

Figura 7.2 Ponto de vista da rede dos níveis de abstrações de serviços de rede, realizados na forma de uma estrutura hierárquica de controle.

Experimentações das definições contidas neste trabalho e no arcabouço do modelo de Serviços Diferenciados foram realizadas para uma situação real de um provedor de serviços a Internet. Onde foram identificadas classes de serviço em suas diversas instâncias, seus níveis de agregação e seleção de um mecanismo de compartilhamento do canal de comunicação entre classes de serviço. Mais especificamente foram utilizados os mecanismos de diferenciação de serviços CBQ e HTB, permitindo a construção de uma hierarquia de controle para a diferenciação de serviços e a realização final do processo de investigação da complexidade inerente a construção de uma rede multi-serviços.

A utilização da sentença, "diferenciação de serviços de rede utilizando um roteador CBQ", no título deste trabalho, resume todas as etapas para a construção de serviços diferenciados. Pois este método foi originalmente concebido para prover serviços diferenciados em uma rede multi-serviços e sua estrutura captura em ultima instancia a realização de todos os níveis de abstração desenvolvidos até aqui.

Portanto, ficou constatada a viabilidade de transformação do modelo Internet em uma plataforma multi-serviço. Permanece, no entanto, a busca pelo desenvolvimento de soluções algorítmicas que possam demonstrar melhor desempenho e adequação as solicitações do cliente por serviços de rede.

Apêndice A

Arquivos de Configuração

A.1 Configuração da interface eth0

A configuração das interfaces para ambos métodos CBQ e HTB foram produzidas através da utilização da ferramenta de configuração de controle de tráfego ***tc*** na forma de scripts dados a seguir.

A.1.1 Método CBQ

```
# adiciona a disciplina de fila raiz, cbq, junto a interface eth0
tc qdisc add dev eth0 root handle 1:0 cbq bandwidth 10Mbit avpkt 1000

# adiciona a classe raiz, cbq, junto a disciplina de fila raiz
tc class add dev eth0 parent 1:0  classid 1:1 cbq bandwidth 10Mbit
rate 512Kbit allot 1514 prio 8 maxburst 20 avpkt 1000

# adiciona a classe correspondente a agência AG1
tc class add dev eth0 parent 1:1  classid 1:2 cbq bandwidth 10Mbit
rate 256Kbit allot 1514 prio 7 maxburst 20 avpkt 1000 bounded

# adiciona a classe correspondente a agência AG2
tc class add dev eth0 parent 1:1  classid 1:3 cbq bandwidth 10Mbit
rate 256Kbit allot 1514 prio 7 maxburst 20 avpkt 1000 isolated borrow

# adiciona a disciplina de fila, folha, da agência AG1
tc qdisc add dev eth0 parent 1:2 sfq quantum 1514b perturb 15

# adiciona a disciplina de fila, folha, da agência AG2
tc qdisc add dev eth0 parent 1:3 sfq quantum 1514b perturb 15
```

```
# adiciona o filtro F1, junto a disciplina raiz, correspondente ao
# agregado de tráfego da classe dos servidores.
tc filter add dev eth0 parent 1:0 protocol ip prio 10 u32 match ip dst
192.168.168.0/26 flowid 1:2

# adiciona o filtro F2, junto a disciplina raiz, correspondente ao
# agregado de tráfego da classe do acesso dedicado.
tc filter add dev eth0 parent 1:0 protocol ip prio 30 u32 match ip dst
192.168.168.64/26 flowid 1:3

# adiciona o filtro F3, junto a disciplina raiz, correspondente ao
# agregado de tráfego da classe do acesso discado.
tc filter add dev eth0 parent 1:0 protocol ip prio 40 u32 match ip dst
192.168.168.128/25 flowid 1:3
```

A.1.2 Método HTB

```
#!/bin/sh
DEV=eth0
DEVIN=hdlc0
case "$1" in
start)
tc qdisc add dev $DEV root handle 1:0 htb default 3
tc class add dev $DEV parent 1:0 classid 1:1 htb rate 512Kbit ceil
512Kbit burst 1k prio 7

tc class add dev $DEV parent 1:1  classid 1:2 htb rate 256Kbit ceil
512Kbit burst 1k prio 5
tc class add dev $DEV parent 1:1  classid 1:3 htb rate 256Kbit ceil
256Kbit burst 1k prio 5

tc qdisc add dev $DEV parent 1:2 pfifo limit 5
tc qdisc add dev $DEV parent 1:3 pfifo limit 5

tc filter add dev $DEV parent 1:0 protocol ip prio 5 handle 1: u32
divisor 1
tc filter add dev $DEV parent 1:0 protocol ip prio 5 u32 match ip dst
200.247.202.0/27 flowid 1:2
tc filter add dev $DEV parent 1:0 protocol ip prio 5 u32 match ip dst
200.247.202.32/27 flowid 1:2
tc filter add dev $DEV parent 1:0 protocol ip prio 5 u32 match ip dst
200.247.202.64/26 flowid 1:3
tc filter add dev $DEV parent 1:0 protocol ip prio 5 u32 match ip dst
200.247.202.128/25 flowid 1:3

tc qdisc add dev $DEVIN handle ffff: ingress
tc filter add dev $DEVIN parent ffff: protocol ip prio 5 u32 match ip
dst 200.247.202.0/24 police rate 512Kbit burst 1 flowid :9

;;
stop)
tc qdisc del dev $DEV root handle 1:0 htb
tc filter del dev $DEVIN parent ffff: protocol ip prio 5 u32 match ip
dst 200.247.202.0/24 police rate 512Kbit burst 1 flowid :9
tc qdisc del dev $DEVIN handle ffff: ingress
```

```
;;
*)
echo usage: `basename $0` \(start\|stop\)
exit 1
;;
esac
```

A.2 Configuração da interface hdlc0

A.2.1 Método CBQ

```
# adiciona a disciplina de fila raiz, cbq, junto a interface hdlc0
tc qdisc add dev hdlc0 root handle 1:0 cbq bandwidth 512Kbit avpkt
1000

# adiciona a classe raiz, cbq, junto a disciplina de fila raiz
tc class add dev hdlc0 parent 1:0  classid 1:1 cbq bandwidth 512Kbit
rate 512Kbit allot 1514 prio 8 maxburst 20 avpkt 1000

# adiciona a classe correspondente a agência AG1
tc class add dev hdlc0 parent 1:1  classid 1:2 cbq bandwidth 512Kbit
rate 256Kbit allot 1514 prio 7 maxburst 20 avpkt 1000 isolated borrow

# adiciona a classe correspondente a agência AG2
tc class add dev hdlc0 parent 1:1  classid 1:3 cbq bandwidth 512Kbit
rate 256Kbit allot 1514 prio 7 maxburst 20 avpkt 1000 bounded

# adiciona a disciplina de fila, folha, da agência AG1
tc qdisc add dev hdlc0 parent 1:2 sfq quantum 1514b perturb 15

# adiciona a disciplina de fila, folha, da agência AG2
tc qdisc add dev hdlc0 parent 1:3 sfq quantum 1514b perturb 15

# adiciona o filtro F1, junto a disciplina raiz, correspondente ao
# agregado de tráfego originado pela classe dos servidores.
tc filter add dev hdlc0 parent 1:0 protocol ip prio 10 u32 match ip
src 192.168.168.0/26 flowid 1:2

# adiciona o filtro F2, junto a disciplina raiz, correspondente ao
# agregado de tráfego originados pela classe do acesso dedicado.
tc filter add dev hdlc0 parent 1:0 protocol ip prio 30 u32 match ip
src 192.168.168.64/26 flowid 1:3

# adiciona o filtro F3, junto a disciplina raiz, correspondente ao
# agregado de tráfego originados pela classe do acesso discado.
tc filter add dev hdlc0 parent 1:0 protocol ip prio 40 u32 match ip
src 192.168.168.128/25 flowid 1:3
```

A.2.2 Método HTB

```
#!/bin/sh
DEVIN=eth0
DEV=hdlc0
case "$1" in
start)
tc qdisc add dev $DEV root handle 1:0 htb default 2
tc class add dev $DEV parent 1:0 classid 1:1 htb rate 512Kbit ceil
512Kbit burst 2k prio 7

tc class add dev $DEV parent 1:1  classid 1:2 htb rate 256Kbit ceil
256Kbit burst 2k prio 5
tc class add dev $DEV parent 1:1  classid 1:3 htb rate 256Kbit ceil
256Kbit burst 2k prio 5

tc qdisc add dev $DEV parent 1:2 pfifo limit 5
tc qdisc add dev $DEV parent 1:3 pfifo limit 5

tc filter add dev $DEV parent 1:0 protocol ip prio 5 handle 1: u32
divisor 1
tc filter add dev $DEV parent 1:0 protocol ip prio 5 u32 match ip src
200.247.202.0/27 flowid 1:2
tc filter add dev $DEV parent 1:0 protocol ip prio 5 u32 match ip src
200.247.202.32/27 flowid 1:2

tc filter add dev $DEV parent 1:0 protocol ip prio 5 u32 match ip src
200.247.202.64/26 flowid 1:3
tc filter add dev $DEV parent 1:0 protocol ip prio 5 u32 match ip src
200.247.202.128/25 flowid 1:3

#tc qdisc add dev $DEVIN handle ffff: ingress
#tc filter add dev $DEVIN parent ffff: protocol ip prio 5 u32 match ip
src 200.247.202.0/24 police rate 512Kbit burst 1 flowid :9
;;
stop)
tc qdisc del dev $DEV root handle 1:0 htb
#tc filter del dev $DEVIN parent ffff: protocol ip prio 5 u32 match ip
src 200.247.202.0/24 police rate 512Kbit burst 1 flowid :9
#tc qdisc del dev $DEVIN handle ffff: ingress
;;
*)
echo usage: `basename $0` \(start\|stop\)
exit 1
;;
esac
```

Apêndice B

Tomada de dados

As coletas de dados foram feitas redirecionando a saída da função de consulta do comando **tc** para um arquivo de log. Este arquivo foi então submetido a um *parser* de forma a extrair somente a quantidade de bytes transmitidos em cada classe de serviço. Os dados extraídos foram então submetidos a uma planilha de cálculo para observação e análise. Sendo que cada tomada de dados corresponde a um período de 20 minutos de observação do canal.

B.1 Tomada de dados da interface *eth0*

B.1.1 Criação do arquivo de log

```
#!/usr/bin/perl -w
#

# interface a ser monitorada
$interface = "eth0";
# arquivo de log onde os dados serao armazenados
$logfile = "/dados_tc/eth0";
# variavel usada como contador das amaostras
$amostras = 0;

# 241 amostras que totalizam 20 minutos
while ($amostras < 241)
{
```

```
if ($interface ne "") {
    # chamada do sistema para colocar o resultado de um comando
    # tc de estatisticas para o arquivo de log
    system "tc -s qdisc show dev $interface >> $logfile &";
    # pausa de 5 segundos
    sleep 5;

    $amostras++;
}

}
```

B.1.2 Amostra do arquivo de *log*

```
qdisc sfq 800a: quantum 1514b perturb 15sec
Sent 3837205 bytes 8014 pkts (dropped 0, overlimits 0)

qdisc sfq 8009: quantum 1514b perturb 15sec
Sent 1216071 bytes 1558 pkts (dropped 0, overlimits 0)

qdisc cbq 1: rate 10Mbit (bounded,isolated) prio no-transmit
Sent 5067107 bytes 9637 pkts (dropped 0, overlimits 0)
borrowed 0 overactions 0 avgidle 624 undertime 0
```

B.2 Tomada de dados da hdlc0

B.2.1 Criação do arquivo de log

```
#!/usr/bin/perl -w
#


$interface = "hdlc0";
$logfile = "/dados_tc/hdlc0";
$amostras = 0;


while ($amostras < 241)
{
if ($interface ne "") {
    system "tc -s qdisc show dev $interface >> $logfile &";
    sleep 5;

    $amostras++;
}

}
```

B.2.2 Amostra do arquivo de log

```
qdisc pfifo 8014: limit 5p
Sent 582777 bytes 1280 pkts (dropped 0, overlimits 0)
backlog 1p

qdisc pfifo 8013: limit 5p
Sent 387517 bytes 562 pkts (dropped 0, overlimits 0)

qdisc htb 1: r2q 10 default 3 dcache 0
deq_util 1/37037 deq_rate 251 trials_per_deq 1
dcache_hits 0 direct_packets 10
Sent 976223 bytes 1857 pkts (dropped 0, overlimits 637)
backlog 1p
```

Referências

Almesberger, Werner. *Linux Traffic Control: Implementation Overview.* ftp://lrcftp.epfl.ch/pub/people/almesber/pub/tcio-current.ps.gz, In Technical Report SSC/1998/037, EPFL, Novembro de 1998.

Bennet, J; Zang, H. *Hierarchical Packet Fair Queuing Algorithms.* In ACM/IEEE Transaction on Networking, Outubro 1997, pp. 675-689.

Bennet, J; Zang, H. *WF2Q: Worst-case fair weighted fair queueing.* In Proceedings of IEEE INFOCOM '96, 1996, pp. 120--128.

Blake S., D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. *An architecture for differentiated services.* RFC 2475, Network Working Group, Dec 1998.

Braun, T. et al. *A Linux Implementation of a Differentiated Services Router.* Sathya Rao and Kaare Ingar Sletta, editors. In Next Generation Networks – Networks and Service for the Information Society, volume 1938 of Lectures Notes in Computers Science, Outubro de 2000, pp. 302-315.

Catania V., G. Ficili, S. Palazzo, D. Panno. "*A Comparative Analysis of Fuzzy versus Conventional Policing Mechanisms for ATM networks*", IEEE/ACM Transactions on Networking, vol. 4, No.3, June 1996.

Demers A., Keshav, S; Shenker. *Analysis and simulation of a fair queueing algorithm.* In Journal of Internetworking Research and experience, , Outubro de 1990, pp. 3-26.

Floyd, Sally; Jacobson, Van. *Link-Sharing and Resource Management Models for packet Networks.* In IEEE/ACM Transaction on Networking, Agosto de 1995, pp. 365-386.

Floyd, S. *Notes on Class-Based Queueing: Setting parameters.* Draft não publicado, Julho de 1995. URL ftp://www-nrg.ee.lbl.gov/ floyd/papers.ht ml.

Guérin, Roch; Peris, V. *Quality-of-service in packet networks: basic mechanisms and directions*. In Computer Networks and ISDN, Special issue on multimedia communications over packet-based networks, 1998.

J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski. Assured Forwarding PHB group. Request For Comments (Proposed Standard) RFC 2597, *Internet Engineering Task Force*, June 1999.

Huston, Geoff. *Internet Performance Survival Guide: Basic Mechanisms and Directions*. Editora John Wiley and Sons. Primeira Edição. Fevereiro de 2000.

IETF Differentiated Services Working Group, DiffServ. Http://www.ietf.org/html.charters/diffserv-charter.html.

Jacobson, Van. *Congestion Avoidance and Control*. Proc. SIGCOMM'88. Agosto de 1988, pp. 314-329.

Kilkki, Kalevi. *Differentiated services for the Internet*. MacMillan Technology Series. Editora New Riders Publishing. Primeira Edição. Junho de 1999.

Nichols, K., Blake, S., Baker, F., and Black, D. *RFC 2474: Denition of the Di erentiated Services Field* (DS Field) in the IPv4 and IPv6 Headers, December 1998.

Nichols K., V. Jacobson and K. Poduri. *Expedited Forwarding PHB Group*. Internet RFC 2598, June 1999.

Parekh, Abhay; Gallager, Robert. *A generalized processor sharing approach to flow control: the single node case*. In ACM/IEEE Transaction on Networking, Junho de 1993, pp. 344-357.

Risso, Fulvio; Gevros, Panos. *Operational and Performance Issues of a CBQ router*. In Computer Communication Review, Outubro de 1999, Volume 29, Numero 5.

Wakeman, Ian et al. *Implementing Real Time Packet Forwarding Policies unsing Streams*. In Usenix 1995 Technical Conference, Janeiro de 1995, New Orleans, Lousiana, pp. 71-82.

Yung, P. *Recursive Estimation and Time Series Analysis*. Springer-Verlag, 1984, pp. 60-65.